CSDN 首页 博客 学院 下载 论坛 APP 问答 商城 活动 VIP会员 续费9折 专题 招聘 ITeye GitChat 图文课 Python工程师 写博客

minigui代码分析

5年06月13日 12:48:09 lidongliang07 阅读数 730

一、minigui运行模式

线程模式：MiniGui-Threads

：_MGRM_THREADS

在MiniGui-Threads上的程序可以在不同的线程中建立多个窗口，但所有的窗口在一个进程或地址空间中运行，传统意义上的嵌入式操作系统。

进程模式：MiniGui-Processes

：_MGRM_Processes或者定义_LITE_VERSION

Gui-Processes上每个程序是单独的进程，每个进程也可以建立多个窗口，并且实现了多进程窗口系统。适用于具有完整UNIX特性的嵌入式式系统。

独立应用模式：MiniGui-Standalone

：_MGRM_STANDALONE 或者定义_LITE_VERSION和_STAND_ALONE

独立任务的方式运行，既不需要多进程支持也不需要多线程支持。

二、数据结构

CreateMainWindow函数参数：PMAINWINCREATE pCreateInfo

体MAINWINCREATE 定义了被创建的窗口的位置、标题、类型等基本参数。实际上包含了创建窗口的UI风格和窗口处理函数两方面的内容。
AINWINCREATE为指向该结构体的指针。

```
typedef struct _MAINWINCREATE
{
    DWORD dwStyle;                                      //窗口风格
    DWORD dwExStyle;                                    //窗口的附加风格
    const char* spCaption;                              //窗口的标题
    HMENU hMenu;                                        //附加在窗口上的菜单句柄
    HCURSOR hCursor;                                    //在窗口中所使用的鼠标光标句柄
    HICON hIcon;                                        //程序的图标句柄
    HWND  hHosting;                                     //窗口消息队列的托管窗口
    int (*MainWindowProc)(HWND, int, WPARAM, LPARAM);   //该窗口的消息处理函数指针，回调函数
    int lx, ty, rx, by;                                 //窗口相对屏幕的绝对坐标，以象素点表示
    int iBkColor;                                       //窗口背景颜色
    DWORD dwAddData;                                    //附带给窗口的一个32 位值
    DWORD dwReserved;                                   //预留，没有用到
}MAINWINCREATE;
typedef MAINWINCREATE* PMAINWINCREATE;
```

MAINWIN结构体：主窗口的详细信息由该结构体给出

```
typedef struct _MAINWIN
{
    /* These fields are similiar with CONTROL struct. */
    unsigned char DataType;     // 数据类型，表示是否是一个窗口（主窗口或者控件窗口），对于该结构和CONTROL结构，都必须是TYPE_HWND</s
    unsigned char WinType;      // 判断是否是主窗口，对于该结构，必须是TYPE_MAINWIN</span>
    unsigned short Flags;       // special runtime flags, such EraseBkGnd flags

    int left, top;      // the position and size of main window.
    int right, bottom;
    int cl, ct;         // the position and size of client area.
    int cr, cb;

    DWORD dwStyle;      // the styles of main window.
    DWORD dwExStyle;    // the extended styles of main window.

    int iBkColor;       // the background color.
    HMENU hMenu;        // handle of menu.
```

一本用漫画写的算法书，小白也能看得懂！ 关闭

```
    HACCEL hAccel;      // handle of accelerator table.
    HICON hIcon;        // handle of icon.
    HMENU hSysMenu;     // handle of system menu.
    PLOGFONT pLogFont;  // pointer to logical font.

    char* spCaption;    // the caption of main window.
    int   id;           // the identifier of main window.

    LFSCROLLBARINFO vscroll; // the vertical scroll bar information.
    LFSCROLLBARINFO hscroll; // the horizital scroll bar information.

    /** the window renderer */
    WINDOW_ELEMENT_RENDERER* we_rdr;

    HDC   privCDC;      // the private client DC.
    INVRGN InvRgn;      // 这个主窗口的无效区域，在处理MSG_PAINT消息时很重要
    PGCRINFO pGCRInfo;  // pointer to global clip region info struct.

    // the Z order node.
    int idx_znode;

    PCARETINFO pCaretInfo; // pointer to system caret info struct.

    DWORD dwAddData;    // the additional data.
    DWORD dwAddData2;   // the second addtional data.

    int (*MainWindowProc)(HWND, int, WPARAM, LPARAM); // the address of main window procedure.

    struct _MAINWIN* pMainWin; // the main window that contains this window. for main window, always be itself.

    HWND hParent;       // the parent of this window.   for main window, always be HWND_DESKTOP.

    /* Child windows.*/
    HWND hFirstChild;    // the handle of first child window.
    HWND hActiveChild;  // the currently active child window.
    HWND hOldUnderPointer;  // the old child window under pointer.
    HWND hPrimitive;    // the premitive child of mouse event.

    NOTIFPROC NotifProc;    // the notification callback procedure.

    /* window element data. */
    struct _wnd_element_data* wed;

    /* Main Window hosting. The following members are only implemented for main window.*/
    struct _MAINWIN* pHosting; // the hosting main window.
    struct _MAINWIN* pFirstHosted; // the first hosted main window.
    struct _MAINWIN* pNextHosted;  // the next hosted main window.

    PMSGQUEUE pMessages;  // the message queue.

    GCRINFO GCRInfo; // the global clip region info struct. put here to avoid invoking malloc function.

#ifdef _MGRM_THREADS
    pthread_t th;        // the thread which creates this main window.
#endif
        //the controls as main
        HWND hFirstChildAsMainWin;

    HDC   secondaryDC;                // the secondary window dc.
    ON_UPDATE_SECONDARYDC update_secdc; // the callback of secondary window dc
    RECT  update_rc;
} MAINWIN;
```

MSGQUEUE消息队列

```
struct _MSGQUEUE
{
    DWORD dwState;              // message queue states

```

一本用漫画写的算法书，小白也能看得懂！
关闭

```
#ifdef _MGRM_THREADS
    pthread_mutex_t lock;       // lock
    sem_t wait;                 // the semaphore for wait message
    sem_t sync_msg;             // the semaphore for sync message
#endif

    PQMSG  pFirstNotifyMsg;     // head of the notify message queue
    PQMSG  pLastNotifyMsg;      // tail of the notify message queue

#ifdef _MGRM_THREADS
    PSYNCMSG pFirstSyncMsg;     // head of the sync message queue
    PSYNCMSG pLastSyncMsg;      // tail of the sync message queue
#else
    IDLEHANDLER OnIdle;         // Idle handler
#endif

#ifdef _MGRM_THREADS
    PMAINWIN pRootMainWin;      // The root main window of this message queue.
#endif

    MSG* msg;                   /* post message buffer */
    int len;                    /* buffer len */
    int readpos, writepos;      /* positions for reading and writing */

    int FirstTimerSlot;         /* the first timer slot to be checked */
    DWORD TimerMask;            /* timer slots mask */

    int loop_depth;             /* message loop depth, for dialog boxes. */
};
```

二、CreateMainWindow函数流程

init_desktop_win

```
<pre name="code" class="objc">static void init_desktop_win (void)
{
    static MAINWIN sg_desktop_win;
    PMAINWIN pDesktopWin;

    LICENSE_SET_MESSAGE_OFFSET();

    pDesktopWin = &sg_desktop_win; //作为默认的Desktop

    pDesktopWin->pMessages         = __mg_dsk_msg_queue;  // 自己的消息队列,其他窗口发送的SendMessage将消息压入该队列[luther.g
    pDesktopWin->MainWindowProc    = DesktopWinProc;      // 桌面默认回调处理函数

    pDesktopWin->DataType          = TYPE_HWND;
    pDesktopWin->WinType           = TYPE_ROOTWIN;
#ifdef _MGRM_THREADS
    pDesktopWin->th                = __mg_desktop;
#endif

    pDesktopWin->pLogFont          = GetSystemFont (SYSLOGFONT_WCHAR_DEF);
    pDesktopWin->spCaption         = "THE DESKTOP WINDOW";

    pDesktopWin->pGCRInfo          = &sg_ScrGCRInfo;
    pDesktopWin->idx_znode         = 0;

    pDesktopWin->pMainWin          = pDesktopWin;
    pDesktopWin->we_rdr            = __mg_def_renderer;

    __mg_hwnd_desktop = (HWND)pDesktopWin;  // 登记到desktop的全局量中
    __mg_dsk_win  = pDesktopWin;
}
```

一本用漫画写的算法书，小白也能看得懂！
关闭

main函数

```
#define main_entry main

#define MiniGUIMain \
MiniGUIAppMain (int args, const char* argv[]); \
int main_entry (int args, const char* argv[]) \
{ \
    int iRet = 0; \
    if (InitGUI (args, argv) != 0) { \
        return 1; \
    } \
    iRet = MiniGUIAppMain (args, argv); \
    TerminateGUI (iRet); \
    return iRet; \
} \
int MiniGUIAppMain
```

CreateMainWindow函数流程

```
HWND GUIAPI CreateMainWindowEx (PMAINWINCREATE pCreateInfo,
        const char* werdr_name, const WINDOW_ELEMENT_ATTR* we_attrs,
        const char* window_name, const char* layer_name)
{
    // 指针
    PMAINWIN pWin;

    if (pCreateInfo == NULL) {
        return HWND_INVALID;
    }

    if (!(pWin = calloc(1, sizeof(MAINWIN)))) { //分配结构体内存
        return HWND_INVALID;
    }

#ifdef _MGRM_THREADS //这是重要部分，用于找到消息队列
    if (pCreateInfo->hHosting == HWND_DESKTOP || pCreateInfo->hHosting == 0) {
        /*如果托管窗口为桌面窗口或者没有托管窗口，为新建的主窗口创建线程信息和消息队列，获取本线程关联的消息队列结构体。
         *若获取失败说明该窗口是最顶层的主窗口，目前还有对应的消息队列，则为其创建一个消息队列结构体*/
        if ((pWin->pMessages = GetMsgQueueThisThread ()) == NULL) { //试图获取本线程关联的消息队列结构体
            if (!(pWin->pMessages = mg_InitMsgQueueThisThread ()) ) { //试图去创建一个消息队列结构体
                free (pWin);

                return HWND_INVALID;
            }
            pWin->pMessages->pRootMainWin = pWin;//如果创建消息队列结构体成功，则设置当前窗口为根窗口
        }
        else {
            /* Already have a top level main window, in case of user have set
               a wrong hosting window */
            pWin->pHosting = pWin->pMessages->pRootMainWin;//设置托管口
        }
    }
    else {
        pWin->pMessages = GetMsgQueueThisThread (); //直接获取，这种情况下，是可以肯定消息队列已经存在
        if (pWin->pMessages != kernel_GetMsgQueue (pCreateInfo->hHosting) || //该函数的调用者必须和hosting的消息队列所在线程
                pWin->pMessages == NULL) {
            free (pWin);

            return HWND_INVALID;
        }
    }

    if (pWin->pHosting == NULL) //如果当前窗口的托管主窗口为空，利用传递的函数参数获得托管主窗口信息
        pWin->pHosting = gui_GetMainWindowPtrOfControl (pCreateInfo->hHosting);
    /* leave the pHosting is NULL for the first window of this thread. */
#else
    pWin->pHosting = gui_GetMainWindowPtrOfControl (pCreateInfo->hHosting); //运行模式为非MiniGui-Threads，获得托管窗口的句柄
```

一本用漫画写的算法书，小白也能看得懂！ 关闭

```
    if (pWin->pHosting == NULL)
        pWin->pHosting = __mg_dsk_win; //托管窗口句柄为空，设置托管窗口为默认桌面窗口

    pWin->pMessages = __mg_dsk_msg_queue; //将消息队列设置为默认桌面消息队列
#endif

    pWin->pMainWin      = pWin; //以下部分在初始化结构体成员，可以忽略
    pWin->hParent       = 0;    //当前窗口的父窗口不存在
    pWin->pFirstHosted  = NULL; //第一个托管主窗口
    pWin->pNextHosted   = NULL; //下一个托管主窗口
    pWin->DataType      = TYPE_HWND;      //数据类型
    pWin->WinType       = TYPE_MAINWIN;   //窗口类型

#ifdef _MGRM_THREADS
    pWin->th            = pthread_self(); //创建主窗口的线程
#endif

    pWin->hFirstChild   = 0;    //第一个子窗口的句柄为0，即不存在
    pWin->hActiveChild  = 0;    //当前活动的子窗口的句柄为0，即不存在
    pWin->hOldUnderPointer = 0; //旧的子窗口
    pWin->hPrimitive    = 0;

    pWin->NotifProc     = NULL;

    pWin->dwStyle       = pCreateInfo->dwStyle;
    pWin->dwExStyle     = pCreateInfo->dwExStyle;

#ifdef _MGHAVE_MENU
    pWin->hMenu         = pCreateInfo->hMenu;
#else
    pWin->hMenu         = 0;
#endif
    pWin->hCursor       = pCreateInfo->hCursor;
    pWin->hIcon         = pCreateInfo->hIcon;

#ifdef _MGHAVE_MENU
    if ((pWin->dwStyle & WS_CAPTION) && (pWin->dwStyle & WS_SYSMENU)) //如果当前窗口包含标题且包含系统菜单，则创建系统菜单
        pWin->hSysMenu= CreateSystemMenu ((HWND)pWin, pWin->dwStyle);
    else
#endif
        pWin->hSysMenu = 0; //否则系统菜单项为0

    pWin->spCaption    = FixStrAlloc (strlen (pCreateInfo->spCaption));//分配空间存放标题
    if (pCreateInfo->spCaption [0]) //如果函数参数结构体的标题字符串的第一个字符非空
        strcpy (pWin->spCaption, pCreateInfo->spCaption); //复制标题到结构体的标题项

    pWin->MainWindowProc = pCreateInfo->MainWindowProc; //消息处理函数
    pWin->iBkColor    = pCreateInfo->iBkColor;

    pWin->pCaretInfo = NULL; //指向系统插入字符信息结构体

    pWin->dwAddData   = pCreateInfo->dwAddData;
    pWin->dwAddData2  = 0;
    pWin->secondaryDC = 0;

    /* Scroll bar */ //下面是初始化滚动条相关的内容
    if (pWin->dwStyle & WS_VSCROLL) { //垂直方向的滚动条
        pWin->vscroll.minPos = 0;
        pWin->vscroll.maxPos = 100;
        pWin->vscroll.curPos = 0;
        pWin->vscroll.pageStep = 101;
        pWin->vscroll.barStart = 0;
        pWin->vscroll.barLen = 10;
        pWin->vscroll.status = SBS_NORMAL;
    }
    else
        pWin->vscroll.status = SBS_HIDE | SBS_DISABLED;

    if (pWin->dwStyle & WS_HSCROLL) { //水平方向的滚动条
        pWin->hscroll.minPos = 0;
        pWin->hscroll.maxPos = 100;
```

0
一本用漫画写的算法书，小白也能看得懂！
关闭

```
        pWin->hscroll.curPos = 0;
        pWin->hscroll.pageStep = 101;
        pWin->hscroll.barStart = 0;
        pWin->hscroll.barLen = 10;
        pWin->hscroll.status = SBS_NORMAL;
    }
    else
        pWin->hscroll.status = SBS_HIDE | SBS_DISABLED;

    /** perfer to use parent renderer */ //初始化渲染器相关的内容，这时可以忽略这一部分
    if (pWin->dwExStyle & WS_EX_USEPARENTRDR) {
        if (((PMAINWIN)pCreateInfo->hHosting)->we_rdr) {
            pWin->we_rdr = ((PMAINWIN)pCreateInfo->hHosting)->we_rdr;
            ++pWin->we_rdr->refcount;
        }
        else {
            return HWND_INVALID;
        }
    }
    else {
        /** set window renderer */
        set_window_renderer (pWin, werdr_name);
    }

    /** set window element data */
    while (we_attrs && we_attrs->we_attr_id != -1) {
        // append_window_element_data (pWin,
        //       we_attrs->we_attr_id, we_attrs->we_attr);
        DWORD _old;
        set_window_element_data ((HWND)pWin,
                we_attrs->we_attr_id, we_attrs->we_attr, &_old);
        ++we_attrs;
    }

    /** prefer to parent font */
    if (pWin->dwExStyle & WS_EX_USEPARENTFONT)
        pWin->pLogFont = __mg_dsk_win->pLogFont;
    else {
        pWin->pLogFont = GetSystemFont (SYSLOGFONT_WCHAR_DEF);
    }

    if (SendMessage ((HWND)pWin, MSG_NCCREATE, 0, (LPARAM)pCreateInfo)) // MSG_NCCREATE表示窗口已经创建但是还没有向系统注册
        goto err;

    /** reset menu size */
    ResetMenuSize ((HWND)pWin);

#ifdef __TARGET_FMSOFT__
    pCreateInfo->lx += __mg_mainwin_offset_x;
    pCreateInfo->rx += __mg_mainwin_offset_x;
    pCreateInfo->ty += __mg_mainwin_offset_y;
    pCreateInfo->by += __mg_mainwin_offset_y;
#endif

    SendMessage ((HWND)pWin, MSG_SIZECHANGING, //开始发生一些消息，让窗口进行一些工作
            (WPARAM)&pCreateInfo->lx, (LPARAM)&pWin->left);
    SendMessage ((HWND)pWin, MSG_CHANGESIZE, (WPARAM)&pWin->left, 0);

    pWin->pGCRInfo = &pWin->GCRInfo;

    if (SendMessage (HWND_DESKTOP, MSG_ADDNEWMAINWIN, (WPARAM) pWin, 0) < 0)//这个很重要：把主窗口发送给Desktop窗口托管，进行管理。
        goto err;

    /*
     * We should add the new main window in system and then
     * SendMessage MSG_CREATE for application to create
     * child windows.
     */
    if (SendMessage ((HWND)pWin, MSG_CREATE, 0, (LPARAM)pCreateInfo)) { //发送一个MSG_CREATE消息，告知应用程序窗口已经创建成功
        SendMessage(HWND_DESKTOP, MSG_REMOVEMAINWIN, (WPARAM)pWin, 0);
        goto err;
    }
```

一本用漫画写的算法书，小白也能看得懂！
关闭

```
<pre name="code" class="objc">
```

```
#ifndef _MGRM_PROCESSES
    screensaver_create();
#endif
    return (HWND)pWin;

err:
#ifdef _MGRM_THREADS
    if (pWin->pMessages && pWin->pHosting == NULL) {
        mg_FreeMsgQueueThisThread ();
    }
#endif

    if (pWin->secondaryDC) DeleteSecondaryDC ((HWND)pWin);
    free (pWin);

    return HWND_INVALID;
}
```

判断传入的参数pCreateInfo是否为空

Case NULL：若参数为空，返回HWND_INVALID

Case NOT NULL：若参数不为空，继续执行2

为PMAINWIN类型的pWin分配内存空间，并判断pWin是否为空

Case NULL：分配空间失败，返回HWND_INVALID

Case NOT NULL：分配空间成功，继续执行3

是否定义_MGRM_THREADS：

3.a定义了_MGRM_THREADS，代表minigui的运行模式为MiniGui-Threads

设置pWin的成员pWin->pMessages和pWin->pHosting

3.b没有定义_MGRM_THREADS，代表minigui的运行模式为非MiniGui-Threads

设置pWin的成员pWin->pMessages和pWin->pHosting

设置pWin的成员。

初始化渲染器相关的内容。

SendMessage ((HWND)pWin,MSG_NCCREATE, 0, (LPARAM)pCreateInfo)

表示该窗口已经创建但是还没有向系统进行注册，当收到这种类型的消息时可以对自己创建的对象进行初始化，但不能创建子窗口，也不能进行绘图。如果函数返回值为非零值，创建的窗口将被

SendMessage ((HWND)pWin, MSG_SIZECHANGING,(WPARAM)&pCreateInfo->lx, (LPARAM)&pWin->left);

指示了将要被更改的窗口的大小，当窗口大小将要发生改变时，该消息会发送给窗口。如果你想要控制窗口改变后的实际位置和大小（窗口改变可能是MoveWindow或者其他函数引起的），你需

_SIZECHANGING作为SendMessage函数的第二个参数，并且通过第二个参数返回位置和大小信息。

SendMessage ((HWND)pWin,MSG_CHANGESIZE, (WPARAM)&pWin->left, 0);

确定改变后的窗口大小。

SendMessage (HWND_DESKTOP,MSG_ADDNEWMAINWIN, (WPARAM) pWin, 0);

把主窗口发送给Desktop窗口托管，进行管理，并绘制窗口。

SendMessage ((HWND)pWin,MSG_CREATE, 0, (LPARAM)pCreateInfo);

发送一个MSG_CREATE消息，告知应用程序窗口已经创建成功。

一本用漫画写的算法书，小白也能看得懂！
关闭

股王经典口诀：“换手率”涨三不追，跌四不压...看完恍然大悟！

南针信息 · 顶新

0

想对作者说点什么

nigui成功移植到ubuntu64位平台 阅读数 5296

:系统ubuntu14LTS64bit,同时在32位ubuntu16.04上经过了测试，官方的所有范例程序都能运行。2.... 博文 来自： 嵌入式Linux底...

niGUI3..0.12安装方法整理 阅读数 3577

GUI3..0.12安装方法整理 博文 来自： Evolution

niGUI源码分析：MiniGUIMain中有... 阅读数 2491

]知道，一般C语言的入口都是main，那么MiniGUIMain函数怎么成为入口呢？在MiniGUI源码includ... 博文 来自： zhanglp的专栏

GUI 几个重要Demo 阅读数 2336

[的定制 #include#include#include"WM.h"#include"FRAMEWIN.h"#include"BUTTON.h"#include"B... 博文 来自： 马上学人工智能

良心帖！看完这份路线，你的 Python 入门基础就差不多了！

Python学习路线免费领~

片机多级菜单编程实现-基于二叉树链表 阅读数 730

·机多级菜单编程实现(ZT)建立一个树状的菜单结构，用链表实现链表中包含：1、指向同级左右菜... 博文 来自： tronteng的专栏

nigui 初始化分析 阅读数 1027

入口点main/main-lite.cmain函数 他的功能是调用InitGUI函数初始化MINIGUI，最终调用用户程序... 博文 来自： yuanbinquan...

niGUI原理分析 阅读数 1905

GUI原理分析一． 概述MiniGUI是广泛应用于嵌入式系统的GUI中间件，支持linux。有线程版本... 博文 来自： lieye_leaves的...

)GUI程序原理分析 阅读数 5371

·行应用程序:主要基于顺序执行结构、以得到执行结果为目的、在执行过程中不需要与用户进行任... 博文 来自： 早起的虫儿灬

gui源代码和分析文档 06-20

ii源代码和分析文档，在vc下可以运行，很方便学习和开发，资料很稀有，希望对学习gui和开发gui的哥们有用，谢谢 下载

揭秘：静脉曲张竟是身体缺了它？饭后吃点它，静脉曲张不再来

京宛协同 · 猎媒

K反编译后代码分析（二） 阅读数 1238

一个continue对应一个back原则（switchwhile结构）在这种形式中，一个contiune一定是对应一个b... 博文 来自： 添翼软件

niGUI实现进度条代码 阅读数 905

Sld:progressbar.c,v1.62004/09/2405:02:15suoweiExp$****Listing25.1****progressbar.c:Samplepro... 博文 来自： jia0511的专栏

nigui综合示例代码

gui 的综合示例代码mde包，给大家分享一下 下载

nigui 3.2.0:基于miniStudio应用TrueType字体的过程(1) 阅读数 1174

]MiniGUI上使用TrueType字体时，在网上找了好多文章，总算是搞定了。不过话说这世界变化快,mi... 博文 来自： 10km的专栏

nigui源码讲解------（一） 阅读数 256

softkeyboard作为参考GUI_main.c staticintInitGui() SKBWindow(HWND_DESKTOP); ... 博文 来自： qq_133585732...

空姐说：男朋友轻松延长40分钟，多吃它，你也行！

康森 · 猎媒

nigui源代码

一本用漫画写的算法书，小白也能看得懂！

关闭

ıes-1.6.10.tar.gz jpegsrc.v6b.tar.gz libminigui-1.6.10.tar.gz libpng_src.tgz mde-1.6.10.tar.gz mg- 下载

ıigui对话框+按键列表demo（加注释） 阅读数 357

lude<stdio.h>#include<stdlib.h>#include<unistd.h>#in… 博文 来自： 白鱼儿的博客

ıiGUI.cfg 阅读数 1713

ıiGUIVer2.0.3/1.6.9#Thisconfigurationfileisforclassicwindowstyle.##Copyright(C)2002~2007Feyn… 博文 来自： jia0511的专栏

ıiGUI编程速查表 阅读数 1272

自己学习MiniGui也快一年了，粗浅的学习了一下，本文是根据《MINIGUI-PROG-GUIDE-V3.0-C… 博文 来自： 逐梦的博客

ıigui 自定义按键 阅读数 1667

按键是在静态框基础上增加了MSG_MOUSEMOVEIN（鼠标移入和移出窗口）和MSG_LBUTTON… 博文 来自： livesflying的博客

早晚开水加一物，七天排尽体内多年湿气，健康又漂亮！神奇！

大商 · 猎媒

ıiGUI 按键添加图片 阅读数 581

先把代码粘在下面了，注释可能不是很清晰。#include<stdio.h>#include… 博文 来自： qq_35313839…

gui界面设计&实体按键驱动 阅读数 1万+

ui界面设计&实体按键驱动本文博客链接:http://blog.csdn.net/jdh99,作者:jdh,转载请注明.环境：主机:… 博文 来自： jdh99的专栏

于ucGUI多任务系统的图形用户界面开发 03-12

其他资源都是免费的，是对于c语言初学者的帮助比较大的，其中有单片机,ARM,数据结构，window编程。我也在学c语言，每当我写完一个… 下载

及菜单实例 多级菜单实例 多级菜单实例下载

开高是因为我搞这个花了一天时间，可能是不是这方面的料，但我在网上真的找不到这样东西，只好硬… 论坛

ıiGUI运行模式 阅读数 467

nux这样的类UNIX操作系统相比，一般意义上的传统嵌入式操作系统具有一些特殊性。举例而言，… 博文 来自： aldxa的专栏

Python应用的六大发展方向！学完python你最想做什么技术岗？

零基础python学习方法，快来挑战！

ıigui环境搭建总结 阅读数 937

一直在搭建minigui开发环境，今天早上终于出来了，写个博客纪念下。搭建这个环境需要至少需要… 博文 来自： a13698709128…

ıiGUI的介绍及安装 阅读数 3586

绍 MiniGUI是一个针对嵌入式设备的高级窗口系统，，图形用户界面支持系统。能够支持包含Linu… 博文 来自： 尺树寸泓

ıiGui下滚动字幕和时钟的实现 阅读数 1919

lude //在caseMSG_INITDIALOG:或者caseMSG_CREATE:下创建计时器 SetTimer(hDlg,IDC_TIM… 博文 来自： SpiritedAway…

ıiGUI源码分析——开始篇 阅读数 1724

GUI即将开源，大家可以关注官方微博 http://weibo.com/fm0minigui 文章中即将提到的很多内容点… 博文 来自： 曾经沧海

ıiGUI 移植到pc和arm开发板全过程详解 及错误解答 阅读数 6052

移植了MiniGUI1.3.3到一个开发板上，中间遇到许多问题，在论坛上发问题，结果也没多少人回我… 博文 来自： liaoxinmeng的…

人工智能怎么学？对于转型的程序员有什么要求？

从国内的招聘网站看不得不说AI的岗位及薪资较优势，但是程序员转型有什么要求？

ıiGUI的消息 阅读数 705

://blog.csdn.net/hiruyue/article/details/11603773 博文 来自： Hello,world!

ıigui源代码

gui源代码下载，最完整的源代码下载 下载

一本用漫画写的算法书，小白也能看得懂！ 关闭

nigui1.6.10的完整源码

gui1.6.10完整源码包，里面有： games-1.6.10.tar.gz jpegsrc.v6b.tar.gz libminigui-1.6.10.tar.gz libpng_s… 下载

niGUI学习的十几个经典例子，非常适合初学者 11-27

个MiniGUI学习的经典例子，都已经在FrameBuffer上面编译，运行通过，对于初学者是很好的资料！！ 下载

nigui显示RAW数据图片 阅读数 1191

BITMAP对象设备无关位图结构，BITMAP是设备相关位图结构，主要区别在像素格式上。如果你从… 博文 来自： 学海无涯

湿气重只需要吃它，排出全身顽固湿气，健康又漂亮

树藤 · 猎媒

niGui 多窗口程序(一) 阅读数 998

NIGUi samplemitwindows*/ /*gcc-ometextmetext.c -lminigui-lpthread-lpng-ljpeg -lmgext*/ #include ... 博文 来自： lbaihao的专栏

nigui图片显示 08-23

用LoadBitmapFromFile函数装载一个BMP文件，然后使用 * FillBoxWithBitmap函数在主窗口上显示该图片。 下载

inigui>GDI双缓冲之道 阅读数 1354

界面编程，常见棘手问题就是出现绘图出现闪屏。这一现象也严重影响用户体验。前段时间刚接触… 博文 来自： duola_rain的专栏

niGUI Messge (一) 阅读数 523

循环是事件驱动的GUI编程基础。在循环中，程序从外部输入设备获取某些事件，比如按键或鼠标… 博文 来自： lbaihao的专栏

niGUI的三种运行模式 阅读数 2391

硬盘时发现有几篇2011年整理的博客躺在那里，删了怪可惜的，发上来保存一下~~年代久远，已… 博文 来自： 尺树寸泓

神奇：月瘦30斤不是梦！饭后喝点“它”一个月赘肉全消失！

九龙 · 猎媒

线程、多进程的MINIGUI程序！ 12-13

个程序中实现了minigui的多线程painter程序，使用TCP/IP协议，当有连接的时候为每个连接fork一个进程。在网络发送数据到painter中，… 下载

niGUI源码分析——MiniGUIMain的奥秘 阅读数 2505

来，通过剖析MiniGUI的最简单的例程，来详细说明MiniGUI程序是如何创建和运行的。这个例程，… 博文 来自： 曾经沧海

游戏引擎开发心得(3) 阅读数 555

天忙，没有更新进度，今天周日，要不上。突然觉得我写的东西太自我为中心，应该把我的心得写… 博文 来自： ID_WinMain的…

niGUI 按键切换图片，背景图片，控件透明。 04-11

实现了MiniGUI的模拟触摸屏，按键切换图片，控件透明，适合新手欢迎下载，评论，讨论。 下载

niGUI更换键盘皮肤 阅读数 338

MiniGUI原来的键盘GUI已经有点不符合现在的需求，所以需要定制一套皮肤，根据我定制皮肤的… 博文 来自： 逐梦的博客

一本用漫画写的算法书，小白也能看得懂！
关闭

nigui/mgncs:利用LoadBitmapFromMem函数对摄像头MJPEG格式图像解码 阅读数 501

与虚拟机有关，在virtualbox虚拟机环境下，即使VIDIOC_S_FMT设置了pixelformat为RGB或YUV… 博文 来自： 10km的专栏

nigui3.0.12全面使用使用ttf字库（完美支持中英文混合显示、subpixel渲染） 06-19

nigui3.0.12全面使用任意ttf字库显示字符（完美支持中英文混合编辑、混合显示，控件及主窗口默认使用ttf字库）做了全面叙述，本文原创… 下载

nigui 3.2.0:通过环境变量修改minigui的运行参数 阅读数 771

行环境中可以通过设置环境变量来改变MiniGUI的一些运行参数。这为应用程序部署提供了很大的… 博文 来自： 10km的专栏

niGUI源码分析--hellowworld(1)：MiniGUIMain中有什么奥秘 阅读数 1万+
-篇： MiniGUI源码分析--开始篇接下来，通过剖析MiniGUI的最简单的例程，来详细说明MiniGUI程… 博文 来自： doon的专栏

nigui对线程库的访问 阅读数 727
卜月要把做好的GUI放到目标板上，本来都是重复过N次的操作了，也没有很在意。但是这次下载到… 博文 来自： coolmoon1973…

如果不想穷一辈子:读懂三不卖七不买是关键,可惜无人知晓
钰安 · 猎媒

niGUI 消息类型分类 阅读数 681
统消息MSG_IDLE------Minigui窗口空闲无事件发生的时候，会一直不停的发送该消息给主窗口MS… 博文 来自： jia0511的专栏

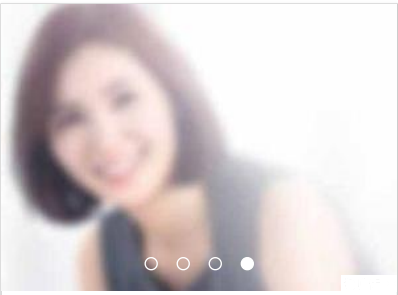
器学习教程 Objective-C培训 交互设计视频教程 颜色模型 设计制作学习

'sql关联查询两次本表 native底部 react extjs glyph 图标 怎么学习互联网大数据 村干部学习大数据心得

lidongliang07 关注

原创	粉丝	喜欢	评论
0	2	1	0

等级： 博客 2 访问： 1万+
积分： 134 排名： 127万+



人脸识别代码

最新文章

线程同步之条件变量：pthread_cond_signal和pthread_cond_wait
gcc和arm-linux-gcc 头文件寻找路径
busybox自带的FTP服务器
MTD应用学习：mtd和mtdblock的区别
理解MiniGUI消息循环和窗口过程

归档

2015年10月 1篇
2015年9月 1篇
2015年8月 1篇
2015年7月 1篇
2015年6月 3篇
展开

热门文章

一本用漫画写的算法书，小白也能看得懂！
关闭

理解MiniGUI消息循环和窗口过程
阅读数 2150

关于busybox自带的ftpd不能使用用户名和密码登陆的解决办法
阅读数 1316

Hi35xx音频（AUDIO）处理模块
阅读数 990

音视频解码模块阅读笔记
阅读数 803

minigui代码分析
阅读数 728

买香港云服务器就选亿速云

低延迟免备案云服务器，高直连稳定独享大带宽香港服速度快稳定有保障

程序人生 CSDN资讯

QQ客服 kefu@csdn.net
客服论坛 400-660-0108
工作时间 8:30-22:00

关于我们 招聘 广告服务 网站地图
百度提供站内搜索 京ICP备19004658号
©1999-2019 北京创新乐知网络技术有限公司

网络110报警服务 经营性网站备案信息
北京互联网违法和不良信息举报中心
中国互联网举报中心 家长监护 版权申诉

0

一本用漫画写的算法书，小白也能看得懂！
关闭