DuiVision

蓝蚂蚁工作室

http://www.blueantstudio.net

[DUIVISION 开发手册]

DuiVision 界面库文档[更新日期：2016-12-16]

目录

DuiVision 开发手册

1. 整体说明

DuiVision 是参考了仿 PC 管家程序、金山界面库、DuiEngine、DuiLib 等多个基于 DirectUI 的界面库开发的。

DirectUI 技术一般是指将所有的界面控件都绘制在一个窗口上，这些控件的逻辑和绘图方式都必须自己进行编写和封装，而不是使用 Windows 控件，所以这些控件都是无句柄的。

DirectUI 技术需要解决的主要问题如下：

1、窗口的子类化，截获窗口的消息。

2、封装自己的控件，并将自己的控件绘制到该窗口上。

3、封装窗口的消息，并分发到自己的控件上，让自己的控件根据消息进行响应和绘制。

4、根据不同的行为发送自定义消息给窗口，以便程序进行调用。

5、一般窗口上控件的组织使用 XML 来描述。

通常 DirectUI 的界面库都采用 XML 配置文件+图片+控制脚本（Lua、Javascript 等）的开发方式，非常类似于 Web 程序的开发方式，当然这里面控制脚本也可以直接使用 C++代码来实现。这种开发方式可以大大提高开发效率，将程序员从繁琐的界面工作中解脱出来，并且通过美工的设计，可以使界面更美观。

2. 开发说明

2.1. XML 资源文件定义

基于 DuiVision 界面库的程序，需要有一个默认的资源定义 XML 文件，此文件默认的位置是 exe 文件所在路径下的 xml\resource.xml 文件 如果使用了 zip 压缩文件来保存所有资源文件，则此文件的位置是在压缩包中的 xml\resource.xml 文件。此文件中可以定义程序的全局配置、XML 文件、字体、图片、文字等资源，示例如下：

```
<?xml version="1.0" encoding="utf-8"?>
<root>

<!--系统配置-->
<res type="cfg" name="defaultStyle" value="" />
<res type="cfg" name="logfile" value="demoui.log" />
<res type="cfg" name="loglevel" value="1" />
<res type="cfg" name="appMutex" value="MUTEX_DUIVISION_DEMO" />
<res type="cfg" name="enableDragFile" value="3" />
```

```
<res type="cfg" name="trayDbClickMsg" value="0" />
<res type="cfg" name="dpiAware" value="0" />

<!--风格设置-->
<res type="style" name="default" value="" />
<res type="style" name="qq" value="qq" />

<!—嵌套的XML资源定义文件-->
<res type="res" lang="zh-cn" file="xml\def_string_zh-cn.xml" />
<res type="res" lang="en-us" file="xml\def_string_en-us.xml" />

<!--XML资源-->
<res type="xml" name="dlg_main" file="xml\dlg_main.xml" />
<res type="xml" name="dlg_skin" file="xml\dlg_skin.xml" />
<res type="xml" name="dlg_about" file="xml\dlg_about.xml" />
<res type="xml" name="dlg_login" file="xml\dlg_login.xml" />
<res type="xml" name="dlg_msgbox" file="xml\dlg_msgbox.xml" />

<!--字体资源-->
<res type="font" lang="zh-cn" name="default" font="微软雅黑" size="12"
bold="false" />
<res type="font" lang="zh-cn" name="big" font="微软雅黑" size="14"
bold="true" italic="false" underline="false" strikeout="false" />

<!--图片资源-->
<res type="img" name="IDB_MAIN_FRAME" file="skins\WindowsBack.png" />
<res type="img" name="IDB_BT_CLOSE" file="skins\BT_CLOSE.png" />
<res type="img" name="IDB_BT_MIN" file="skins\BT_MIN.png" />
<res type="img" name="IDB_BT_MENU" file="skins\BT_MENU.png" />
<res type="img" name="IDB_BT_SKIN" file="skins\BT_SKIN.png" />

<res type="img" name="IDB_TAB_1" file="skins\Tab1.png" />
<res type="img" name="IDB_TAB_2" file="skins\Tab2.png" />

<res type="img" name="IDB_ICON_INFO" file="skins\info.png" />
<res type="img" name="IDB_ICON_WARN" file="skins\warning.png" />
<res type="img" name="IDB_ICON_ERROR" file="skins\error.png" />

<res type="img" name="IDB_MENU_UPDATE" file="skins\MENU_UPDATE.png" />

<!--字符串资源-->
<res type="str" lang="zh-cn" name="APP_NAME" value="DUI测试程序" />
```

```
<res type="str" lang="zh-cn" name="APP_VER" value="1.0.0.1" />
<res type="str" lang="zh-cn" name="OK" value="确定" />
<res type="str" lang="zh-cn" name="CANCEL" value="放弃" />
<res type="str" lang="zh-cn" name="LOGIN" value="登录" />

</root>
```

这些定义的说明如下：

1、全局配置定义

Xml 的 type 是 cfg，目前支持的配置如下：

logfile – 日志文件名，是相对 exe 的路径的文件名，如果未定义，则不会生成日志文件

loglevel – 日志级别，1 表示调试级别，2 表示信息级别，4 表示错误级别，8 表示致命级别

defaultStyle – 默认的风格，resource.xml 中的每个资源定义都可以加一个 style 属性，通过 style 属性指定这条资源定义是针对哪种风格的，只有和当前的风格相同的资源或者默认风格的资源才会被加载，资源定义中指定风格的例子如下：

```
<res type="xml" style="qq" name="dlg_main" file="xml\dlg_wnd.xml" />
```

appMutex – 应用程序互斥量的名字，如果指定了此变量，则应用程序只能创建一个运行的进程，第二个进程运行时候判断如果存在此名字的互斥量，则退出

enableDragFile – 是否允许拖拽一个图片文件到程序窗口来指定当前使用的背景图片

注 :值为 0 不允许拖拽 ,值为 1 只允许控件接收拖拽 ,值为 2 只允许拖拽图片指定背景图片，值为 3 允许控件接收拖拽，也允许拖拽图片指定背景图片。

trayDbClickMsg – 在托盘图标双击是否发送消息，用于重新定义托盘图标双击的行为，托盘图标双击的默认行为是打开程序的主窗口，如果此变量设置为 1，则双击托盘图标会给应用程序发送一个 DUI 消息，消息类型为 MSG_TRAY_DBCLICK，消息的发送方 ID 和名字分别是 TRAY_ICON 和 NAME_TRAY_ICON，可以在 DUI 消息处理类中增加对此消息的处理，来实现自定义的双击动作

2、风格定义

type 是 style，name 是风格的名字，仅用于说明，value 是风格的值，每个资源定义中的 style 对应的是风格的 value 部分。

资源 XML 中所有的定义都可以加一个 style 属性来指定当前定义是针对特定风格的，如果和当前的风格一致，则使用此定义覆盖之前的未指定风格的同名的定义。

例如下面的两个定义是定义的名为 IDB_SCROLL_V 的图片资源，一个是缺省定义，一个是针对 qq 风格的定义，如果当前不是 qq 风格，就会用前一个定义，如果当前是 qq 风格，就会

用针对 qq 风格的定义，如果当前是 qq 风格，但没有第二行针对 qq 风格的定义，则也会使用第一行的缺省定义。

```
<res type="img" name="IDB_SCROLL_V" file="skins\default\SCROLL_V.png"
/>
<res type="img" style="qq" name="IDB_SCROLL_V"
file="skins\qq\SCROLL_V.png" />
```

3、资源文件定义

Type 是 res，用于加载嵌套的资源定义文件，file 是对应的文件路径名，是相对 exe 所在的路径，通过 lang 属性可以指定此资源文件仅针对哪种语言。

4、xml 文件定义

Type 是 xml，name 是其他地方引用时候的名字，file 是对应的文件路径名，是相对 exe 所在的路径。

5、字体定义

Type 是 font，lang 表示是针对哪种语言的字体。其他属性说明：

name – 字体定义名，在其他地方用名字进行引用

font – 字体名

size – 文字大小

bold – 是否粗体（true|false）

italic– 是否斜体（true|false）

underline – 显示下划线（true|false）

strikeout – 显示删除线（true|false）

os – 表示此字体定义适用于哪些操作系统，例如 os="winxp,win7"表示此定义仅针对 xp 和 win7 操作系统，可用的操作系统名字符串包括 win98、winme、winnt、win2000、winxp、win2003、vista、win7、win8。

6、图片资源定义

Type 是 img，name 是其他地方引用时候的名字，file 是对应的文件路径名，是相对 exe 所在的路径。

7、字符串资源定义

Type 是 str，lang 表示是针对哪种语言的字体，name 是其他地方引用时候的名字，value 是字符串内容。

定义的字符串资源可以在各个空间的 title 属性中引用，引用时候要用[]包围，例如 title=”[APP_NAME]”就可以在 title 中引用 APP_NAME 对应的字符串。

8、资源中的多语言定义

资源定义 xml 中每一项都可以指定语言，如果定义中有 lang 属性，则表示这条定义是针对哪种语言的，如果和当前语言不相符，则不会被加载。

2.2. XML 对话框文件定义

程序中所有界面都是基于对话框或菜单等窗口的，每个对话框都需要有一个 XML 定义文件，用于描述对话框中的内容，对话框中主要是组成对话框的各个控件的定义，对话框的 XML 定义示例如下：

```
<?xml version="1.0" encoding="utf-8"?>
<dlg name="dlg_about" title="MsgBox" width="450" height="230" appwin="1"
resize="1" translucent="245" frame="" bkimg="skin:SKIN_PIC_7"
crbk="000000" >
    <base>
        <imgbtn name="button.close" pos="-45,0,-0,29"
skin="IDB_BT_CLOSE" show="1"/>
        <text name="title" crtext="FFFFFF" crmark="800000" font="big"
              pos="10,5,200,25" title="关于[APP_NAME]"
mask="[APP_NAME]" response="0" show="1" />
    </base>
    <body>
        <area name="area-1" pos="0,0,-0,40" begin-transparent="100"
end-transparent="30" />
        <area name="area-2" pos="0,40,-0,-0" begin-transparent="30"
end-transparent="30" />
        <area name="area-3" pos="0,-37,-0,-36" begin-transparent="70"
end-transparent="70" />
        <area name="area-4" pos="0,-36,-0,-0" begin-transparent="88"
end-transparent="88" />
        <img name="icon" pos="25,45" width="128" height="128"
image="skins\scriptnet.jpg" mode="normal" framesize="0" response="0"
show="1" />
        <text crtext="000000" pos="170,45,-25,65" title="[APP_NAME]
[APP_VER]" />
```

```
        <text crtext="000000" pos="170,65,-25,85" title="2013-2014" />
        <linkbtn name="linkbtn1" crtext="800000"
            pos="170,100,-25,130" show="1"
title="http://www.blueantstudio.net"
href="http://www.blueantstudio.net" />
        <button name="button.ok" skin="IDB_BT_DEFAULT" title="[OK]"
    pos="-100,-30,-20,-6" show="1" />
    </body>
</dlg>
```

其中由几部分组成，dlg 标签是对话框自身一些属性的描述，可以设置对话框的大小、背景图片、蒙版图片、透明度、应用程序窗口属性、改变大小属性等；

base 标签下面的内容都是属于对话框的基础控件，一般可以把对话框的标题和关闭按钮等放在基础控件部分定义；

body 标签下面的内容是属于对话框的普通控件，除了基础控件之外，其他内容都放在 body 标签下面定义。

base 和 body 下面都是具体控件的定义描述，可以参考后面关于每个控件的属性说明。

以上对话框 XML 定义的界面效果如下：

2.3. XML 菜单文件定义

菜单也是通过 XML 文件来定义，菜单 XML 的定义示例如下：

```
<?xml version="1.0" encoding="utf-8"?>
<menu title="TrayMenu" width="250" item-height="25" left="30"
    frame-width="0" top-height="72" bottom-height="30"
    bkmode="mid" width-lt="5" height-lt="70" width-rb="5" height-rb="30"
    bkimg="skins\menu\360TrayMenu_218.png" >

    <text pos="10,5,-10,25" crtext="FFFFFF" font="bold" title="DUI托盘
```

```
菜单" />
    <img pos="-60,5" width="48" height="48"
image="skins\icon\Movies.png"
        mode="normal" framesize="1" tip="图片" />
    <text pos="10,-25,-10,-5" crtext="808080" font="bold" title="蓝蚂蚁
工作室" />

    <menuitem name="restore_mainwnd" skin="" image="" title="恢复窗口"
font="bold" action="show-window:dlg_main" />
    <menuitem separator="1" skin="IDB_MENU_SEP" />
    <menuitem name="menu_main" menu="menu_main" />
    <menuitem name="menuitem.sub" title="子菜单" width="150"
skin="IDB_MENU_ARROW"
        bkmode="frame" bkimg="skin:IDB_MENU_BACK" frame-width="3"
top-height="0" bottom-height="0" >
        <menuitem name="360safe" title="360安全卫士"
action="dlg:dlg_login" />
        <menuitem name="360sd" title="360杀毒" action="dlg:dlg_login" />
        <menuitem name="menuitem.360.sub1" title="360工具" width="150"
skin="IDB_MENU_ARROW"
            bkmode="frame" bkimg="skin:IDB_MENU_BACK" frame-width="3"
top-height="0" bottom-height="0" >
            <menuitem name="360driver" title="360驱动修复"
action="dlg:dlg_login" />
            <menuitem name="360soft" title="360软件管家"
action="dlg:dlg_login" />
        </menuitem>
    </menuitem>
    <menuitem name="menuitem.runtest" title="执行进程"
action="run:{platpath}DuiVisionDemo.2008d.exe|testcmd" />
    <menuitem separator="1" skin="IDB_MENU_SEP" />
    <menuitem name="item_help" skin="IDB_MENU_HELP" img-count="3"
title="帮助" action="link:http://www.blueantstudio.net" />
    <menuitem separator="1" skin="IDB_MENU_SEP" />
    <menuitem name="close_app" skin="" image="" title="退出"
action="close-window:dlg_main" />

</menu>
```

菜单定义文件的根节点是 menu 节点，menu 节点可以定义一些菜单的属性，包括菜单的大小、图标区域的宽度、背景图片等。

menu 节点下面是每个菜单项和控件的定义，菜单项一般用 menuitem 标签进行定义，除了

菜单项之外，还可以定义菜单分隔线，以及其他的控件，例如图片控件。

菜单的显示区域可以分为 3 段 ,中间是菜单项部分 ,上面和下面是可以显示其他控件的区域，如果要在底部区域显示图片、文字等控件 , 这些控件的位置定义中的 Y 坐标都应该使用负数来定义。

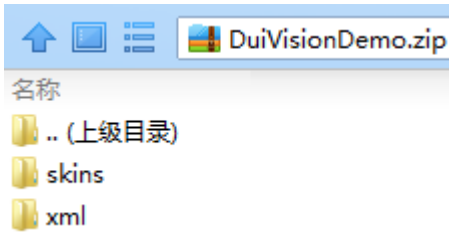
以上菜单定义文件的实际显示效果如下：

2.4. 从 ZIP 压缩文件中加载资源

DuiVision 支持将所有的图片和 XML 资源文件放在一个 zip 格式的压缩文件中，如果使用 zip 格式的资源文件，需要在主程序代码中初始化部分指定使用的压缩文件的文件名。

如果使用 zip 资源文件，则 resource.xml 文件的位置默认是放在 zip 文件中的 xml 子目录下。

建议 zip 文件按照 xml、skins 这样的子目录来压缩，见下面的压缩文件示例：



有 zip 资源文件的情况下，资源文件的加载并不一定是加载的 zip 文件中的内容，加载的优先级如下：

1) 如果只有 zip 压缩文件，没有非压缩的 xml 和 skins 目录，则只会加载 zip 文件中的内容；

2 如果 zip 压缩文件和非压缩的 xml 和 skins 目录同时存在 则优先加载非压缩的 xml 和 skins 目录中的文件，对应的文件不存在的情况下才去 zip 文件中查找是否存在并加载。

之所以这样定义是方便通过非压缩的文件替换压缩文件中部分内容 ,以及方便调试和发布工程，调试阶段可以直接修改非压缩的目录中文件，不用每次修改之后都要再打一次压缩包。

说明：zip 资源文件中仅支持包含 xml、png、bmp 类型的文件，其他文件无法加载，如果有其他类型的文件，请不要放在 zip 文件中，应该单独放在外部目录中加载。

2.5. 将资源 ZIP 压缩文件作为 EXE 的内嵌资源

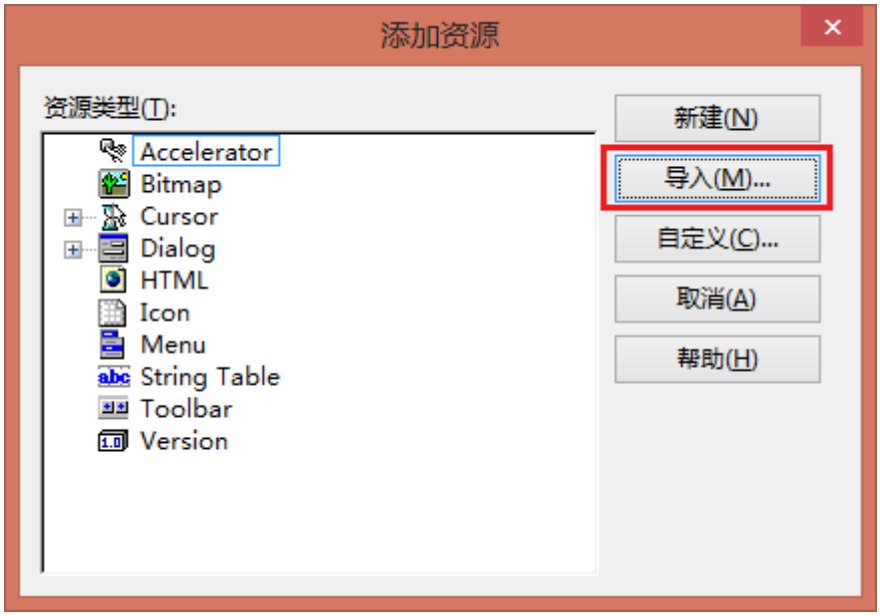
DuiVision 支持将资源 zip 文件作为 EXE 的内嵌资源，将资源文件编译到 exe 中可以生成不依赖任务资源文件，可以独立运行的 exe 文件。

制作内嵌 ZIP 资源的方法：

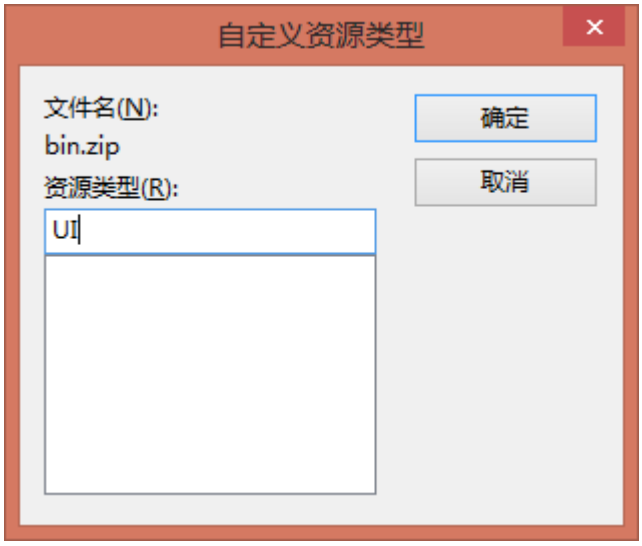
1、将 bin 目录中的所有资源使用的子目录和文件进行压缩为 ZIP 文件(参考上一节压缩时候的目录结构)；

2、将压缩文件作为资源添加到工程的资源中，添加时候资源类型一定要输入“UI”；

在 VC 工程的资源中添加资源，然后选择导入：



在导入对话框中选择资源 zip 文件，在弹出的自定义资源类型对话框中输入 UI，必须使用这种类型，因为 DuiVision 中限定了加载的内嵌界面压缩包资源必须是"UI"类型的。



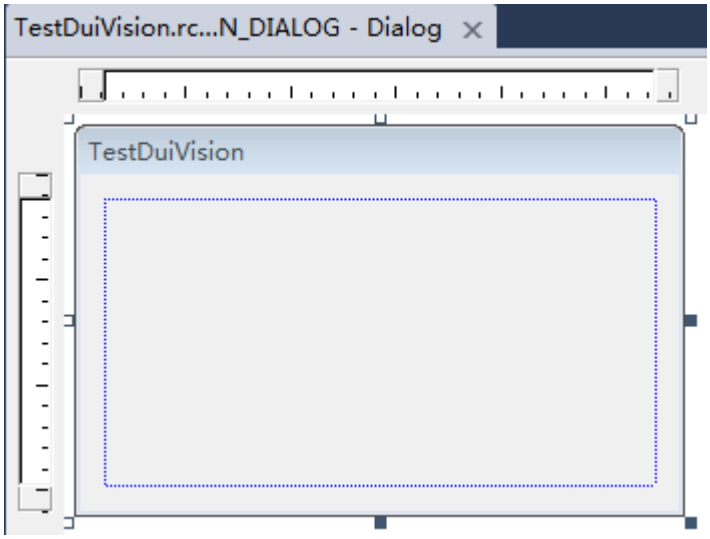
3、调用 DuiSystem 构造函数时候 strResourceFile 参数使用 res:resid 的格式，其中 resid 为已经添加的资源的 ID，资源 ID 可以在 resource.h 文件中查到；

4、这样编译的 EXE 文件就可以使用内嵌的界面资源文件（如果 EXE 所在目录中有资源相关的目录和文件，还是会优先使用外部文件）。

2.6. 创建主程序的方法（人工创建）

1、创建一个基于 DuiVision 的界面程序是比较简单的，在 VC 中创建一个 MFC 对话框工程，可以使用 Unicode 或多字节方式：

工程创建之后，需要将默认对话框资源中的几个按钮和文字都删除，变成一个干净的对话框资源：



2、设置 DuiVision 的头文件和 lib 文件目录

将 DuiVision 的头文件和 lib 文件放在某个位置，并在工程的头文件和 lib 文件路径定义部分添加相应的目录。

然后在 stdafx.h 文件中添加如下几行对 DuiVision 的头文件和 lib 文件的引用。

```
#include "DuiVision.h"
```

Lib 库采用在工程设置中添加引用库文件的方式，针对 VC2008 的 DuiVision 库文件分别是 DuiVision.2008d.lib 和 DuiVision.2008.lib。

添加 lib 引用之后应该就可以编译代码了。

3、DuiVision 库的初始化以及主窗口的定义

在主程序的 App 类 InitInstance()函数中添加 DuiVision 库的引用代码，示例代码如下：

```
    // 初始化DuiVision界面库,可以指定语言,dwLangID为表示自动判断当前语言
    // 1116是Demo程序的应用程序ID，每个DUI应用程序应该使用不同的ID
    // ID主要用于进程间通信传递命令行时候区分应用
    // IDD_DUIVISIONDEMO_DIALOG是工程中创建的那个对话框资源ID，所有窗口都是共用此ID的
    // 第三个参数可以指定资源文件名，如果不指定默认使用xml\resource.xml的定义进行资源的加载，资源文件名的定义方式参考下面的说明
    DWORD dwLangID = 0;
    new DuiSystem(m_hInstance, dwLangID, _T("DuiVisionDemo.ui"), 1116, IDD_DUIVISIONDEMO_DIALOG, "");
```

```
// 创建主窗口
CDlgBase* pMainDlg = DuiSystem::CreateDuiDialog(_T("dlg_main"), NULL, _T(""), TRUE);
// 给主窗口注册事件处理对象
CDuiHandlerMain* pHandler = new CDuiHandlerMain();
pHandler->SetDialog(pMainDlg);
DuiSystem::RegisterHandler(pMainDlg, pHandler);

// 初始化提示信息窗口
DuiSystem::Instance()->CreateNotifyMsgBox(_T("dlg_notifymsg"));

// 按照非模式对话框创建主窗口,可以设置为默认隐藏
pMainDlg->Create(pMainDlg->GetIDTemplate(), NULL);
//pMainDlg->ShowWindow(SW_HIDE);
INT_PTR nResponse = pMainDlg->RunModalLoop();

// 如果是按照模式对话框运行主窗口,只要改为如下代码就可以
//INT_PTR nResponse = pMainDlg->DoModal();

// 释放DuiVision界面库的资源
DuiSystem::Release();
```

如果已经定义了主窗口的 XML 定义文件，添加上面的代码之后应该就可以创建出主窗口了。这段代码做的事情主要是：

1、DuiVision 库的初始化，并指定资源文件的位置（不指定则使用默认的位置）

2、根据 dlg_main 定义加载主窗口界面

3、创建主窗口的事件处理对象，并注册给主窗口（注册之后主窗口的事件都会自动发送给此事件处理对象的 OnDuiMessage 处理函数进行处理）

4、显示主窗口，使用非模态方式运行主窗口的消息循环

5、主界面关闭之后进行 DuiVision 库的释放

注意运行之前要把图片、xml 定义都放在对应的位置，默认图片都在 skins 目录，xml 定义文件都在 xml 目录。

DuiSystem 构造函数第三个参数（资源文件名）的说明，资源文件名可以用下面几种格式：

1、不指定（为空），则使用默认的资源文件名 xml\resource.xml，表示加载 exe 路径下的 xml\resource.xml 文件，根据此文件加载其余的资源；

2、指定 xml 文件，例如 xml\resource.xml，表示加载指定的资源 xml 文件，根据此 xml 文件

加载其余的资源，如果指定了 xml 文件路径则使用绝对路径，如果没有指定 xml 路径，则查找 exe 路径和 exe 下面的 xml 路径看有没有此文件，有的话就加载 xml 文件；

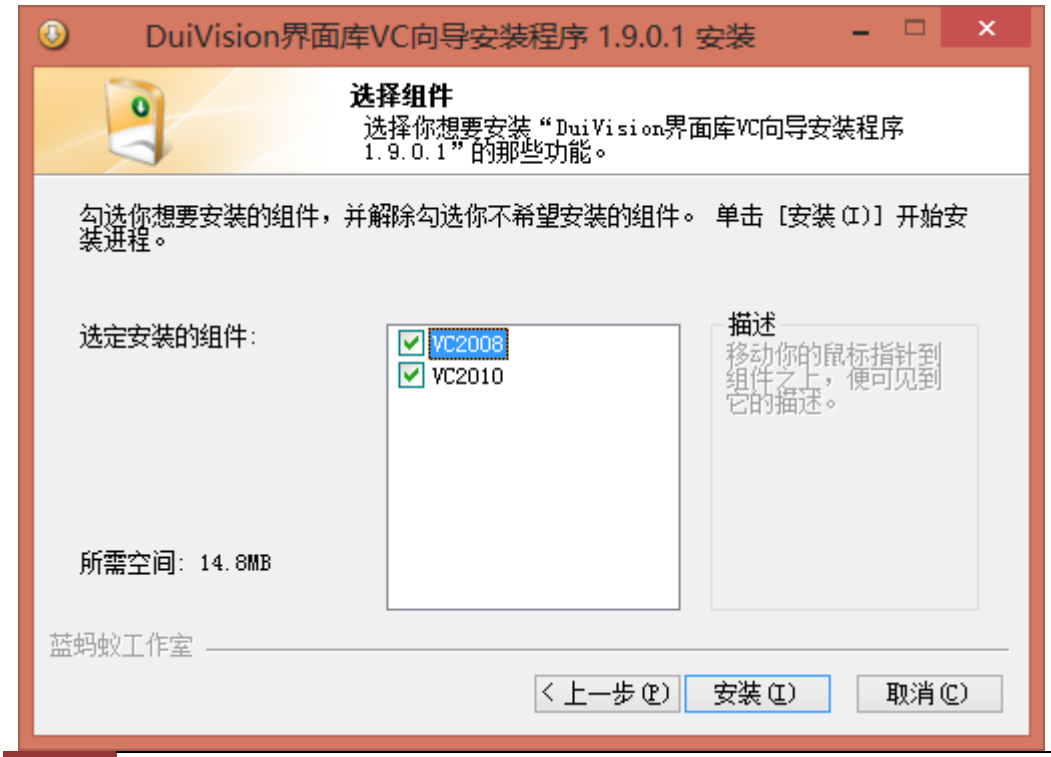
3、指定后缀是.ui 的文件，后缀是.ui 的文件表示是 zip 压缩文件，则首先加载此 zip 压缩文件到内存中，然后再从 zip 压缩文件中解压出相对路径是 xml\resource.xml 的文件进行加载，后续其余文件的加载过程和上面两种情况是相同的，如果 zip 压缩文件已经加载了，则加载其余文件的时候，会先查找物理路径下有没有文件，有的话就优先加载，物理路径下没有找到文件，则查找内存中的 zip 压缩文件中有没有文件，有的话就解压到内存并加载，加载内存文件时候会对内存文件进行缓存，下一次再加载相同的文件时候直接用内存缓存，不需要再进行解压缩；

4、指定前缀是 res:的文件，表示从程序的资源中首先获取到资源 zip 压缩文件，后续的流程和.ui 方式的 zip 压缩文件是相同的，这种情况下，参数 res:后面跟的是程序中的资源 ID，例如 res:1116 表示从程序资源中获取资源 ID 是 1116 的资源进行解压缩。

2.7. 创建主程序和插件的方法（通过向导创建）

DuiVision 提供了创建工程的 VC 向导，目前支持 VC2008、VC2010、VC2013、VC2015，也可以稍微修改之后用于其他的 VC 版本，向导有两种安装方式：

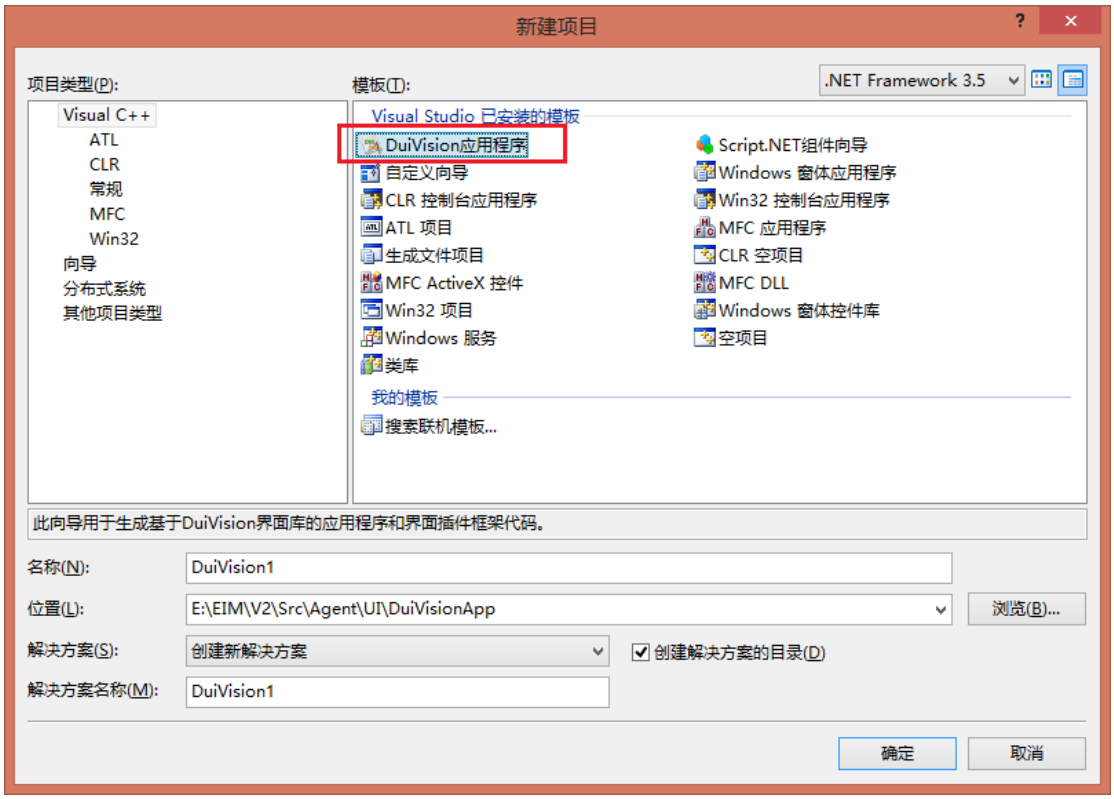
1）从蓝蚂蚁工作室网站或 QQ 群下载安装 VC 向导，VC 向导发布了通过安装包自动安装的版本和人工安装的压缩包版本。如果使用向导安装包进行安装，可以选择安装到哪些 VC 版本中，选择界面如下：



说明：向导的安装程序已经没有更新了，网站上也已经不发布向导安装包，请使用下面的第二种安装方式。

2）**【推荐方法】**更新 github 库，找到其中的 DuiWizard\SetupWizard.js 文件，运行这个 javascript 脚本就可以进行向导的安装，运行时候会提示是否安装 VC2008、VC2010、VC2013、VC2015 版本的向导，这种方式安装向导的优点是如果 DuiVision 版本升级了，只需要更新 github 库，不需要再次安装向导，生成的工程就自动是最新版本了，因为这种方式生成的工程代码是直接从下载的代码目录下复制的。

向导安装之后，在 VC 中新建工程，可以看到模板中多了一个 DuiVision 应用程序模板：



选择此模板就可以创建 DuiVision 工程，创建工程一共有四个步骤，第一步是设置应用程序标题和版权信息：

第二步是选择创建的工程类型，有两种类型，DuiVision 主程序和 DuiVision 插件：

第三步是设置界面中多个 Tab 页情况下每个 Tab 页的标题和事件处理类名字，如果不设置任何 Tab 页，也会自动创建一个默认 Tab 页，注意添加的 Tab 页必须勾选添加列的选择框才会生成对应的代码：

第四步是工程的一些选项设置，如果是新建的一个解决方案工程，则默认会将 DuiVision 库的代码和资源文件拷贝到新建的解决方案目录下，如果是在现有解决方案中新建一个工程，则默认不拷贝，你也可以修改这个选项：

生成的工程中包含 DuiVision 库、工程代码、bin 目录和 bin 目录下的资源文件，针对创建的每个 Tab 页，都会在 bin 目录下的 xml 目录下生成对应的 Tab 页 xml 文件框架，DuiVision 应用程序的 xml 文件在 xml\app 目录下，DuiVision 插件程序的 xml 文件在 xml\plugin 目录下。向导中包含的 DuiVision 库可能不是最新的库，可以在工程生成之后，将工程目录下的 DuiVision 目录和 bin 目录下的一些基础资源文件替换为最新的文件。

2.8. 创建对话框

DuiVision 的对话框实际上就是 MFC 的对话框，对话框分为模态对话框和非模态对话框，模态对话框显示时候不能操作程序中的其他对话框界面，非模态对话框则无此限制。一般程序的主窗口是用的非模态对话框，并且会执行一个消息循环函数，程序中其他的非模态对话框是不需要执行消息循环函数的，只要显示出来就可以。

DuiSystem 封装了常用的创建、删除、查询对话框的操作：

```
static CDlgBase* CreateDuiDialog(LPCTSTR lpszXmlTemplate, CDuiObject* pParentObject,
CString strName = _T(""), BOOL bModule = TRUE, UINT nIDTemplate = 0, BOOL bAdd = TRUE);
static int ShowDuiDialog(LPCTSTR lpszXmlTemplate, CDuiObject* pParentObject, CString
```

```
strName = _T(""), BOOL bModule = TRUE);
static int DuiMessageBox(CDuiObject* pParent, LPCTSTR lpszText, LPCTSTR lpszCaption = _T(""),
UINT uType = MB_OK|MB_ICONINFORMATION, int nWidth = 0, int nHeight = 0);
void AddDuiDialog(CDlgBase* pDuiDialog);
void RemoveDuiDialog(CDlgBase* pDuiDialog);
CDlgBase* GetDuiDialog(int nIndex);
CDlgBase* GetDuiDialog(CString strName);
```

一般模态对话框的使用方法如下：

```
CDlgBase* pDlg = DuiSystem::CreateDuiDialog(strDlgXmlContent, NULL, _T(""), TRUE, 0, TRUE);
if(pDlg != NULL)
{
    int nResponse = pDlg->DoModal();
    DuiSystem::Instance()->RemoveDuiDialog(pDlg);
}
```

CreateDuiDialog 的第 4 个参数为 TRUE，创建之后，调用对话框的 DoModal 函数进行模态显示。

非模态对话框的使用方法如下：

```
m_pNotifyMsgBox = CreateDuiDialog(lpszXmlTemplate, NULL, strName, FALSE);
if(m_pNotifyMsgBox != NULL)
{
    m_pNotifyMsgBox->ShowWindow(SW_SHOW);
}
```

CreateDuiDialog 的第 4 个参数为 FALSE，创建之后，调用对话框的 ShowWindow 函数将窗口显示出来。

2.9. 事件处理类编写

除了界面的描述之外，最主要的工作就是业务逻辑的处理，为了将业务逻辑和界面展示能够更好的分离，DuiVision 中定义了事件处理基类，所有的业务逻辑都应该写在派生的事件处理类中，并把事件处理对象注册到相应的对话框或控件上，这样对应的子控件有事件需要处理的时候，就会自动调用注册的事件处理对象的相应函数。事件处理类只要在处理函数中根据控件的 ID 或名字决定该做什么事情，写相应的处理代码就可以。事件处理类中同时提供了一些函数方便根据 ID 或名字获取到对应的控件对象，并对控件进行操作，例如改变控件文字、获取控件的某个状态等。

事件处理基类是 CDuiHandler，这个类的定义如下，提供了获取控件对象、设置控件的一些参数的函数，方便在事件处理类中对控件的操作：

```
class CDuiHandler
```

```
{
public:
	CDuiHandler(void);
	virtual ~CDuiHandler(void);

	void SetDuiObject(CDuiObject* pDuiObject);
	CControlBase* GetControl(UINT uControlID);
	CControlBase* GetControl(CString strControlName);
	CDlgBase* GetControlDialog(UINT uControlID);

	void SetVisible(CString strControlName, BOOL bIsVisible);
	void SetDisable(CString strControlName, BOOL bIsDisable);
	void SetTitle(CString strControlName, CString strTitle);
	CString GetTitle(CString strControlName);

	virtual void OnInit();
	virtual LRESULT OnDuiMessage(UINT uID, CString strName, UINT Msg, WPARAM wParam,
LPARAM lParam);
	virtual void OnTimer(UINT uTimerID, CString strTimerName);

protected:
	CDuiObject*	m_pDuiObject;			// 关联的DUI对象
};
```

下面这段代码是在派生的事件处理类的 OnDuiMessage 函数中判断如果点击了某个按钮，就修改一个进度条进度的代码：

```
	if(strName == _T("button_normal_3"))
	{
		CDuiProgress* pControl = (CDuiProgress*)GetControl(_T("progress"));
		if(pControl && (Msg == BUTTOM_UP))
		{
			static int g_nProgress = 0;
			g_nProgress += 10;
			if(g_nProgress > 100)
			{
				g_nProgress = 0;
			}
			pControl->SetProgress(g_nProgress);
		}
	}
```

为了简化事件处理类的编写，使事件处理的代码看起来更清晰一些，DuiHandler.h 中定义了一些事件处理函数的消息映射宏，如下表所示：

宏	参数	说明
DUI_DECLARE_MESSAGE_BEGIN	类名	事件处理类的消息映射宏开始
DUI_DECLARE_MESSAGE_END	无	事件处理类的消息映射宏结束
DUI_CONTROL_ID_MESSAGE	控件 ID、处理函数	根据控件 ID，执行相应的处理函数
DUI_CONTROL_IDMSG_MESSAGE	控件 ID、消息、处理函数	根据控件 ID 和消息，执行相应的处理函数
DUI_CONTROL_NAME_MESSAGE	控件名、处理函数	根据控件名，执行相应的处理函数
DUI_CONTROL_NAMEMSG_MESSAGE	控件名、消息、处理函数	根据控件名和消息，执行相应的处理函数

实际定义的样例如下：

```
// 消息处理定义
DUI_DECLARE_MESSAGE_BEGIN(CDuiHandlerMain)
    DUI_CONTROL_ID_MESSAGE(APP_IPC, OnDuiMsgInterprocess)
    DUI_CONTROL_NAME_MESSAGE(NAME_SKIN_WND, OnDuiMsgSkin)
    DUI_CONTROL_NAMEMSG_MESSAGE(NAME_TRAY_ICON, MSG_TRAY_DBCLICK,
OnDuiMsgTrayIconDClick)
    DUI_CONTROL_NAMEMSG_MESSAGE(L"notify_button_1", BUTTOM_UP, OnDuiMsgNotifyButton1)
    DUI_CONTROL_NAMEMSG_MESSAGE(L"notify_button_2", BUTTOM_UP, OnDuiMsgNotifyButton2)
    DUI_CONTROL_NAMEMSG_MESSAGE(L"notify_button_3", BUTTOM_UP, OnDuiMsgNotifyButton3)
    DUI_CONTROL_NAMEMSG_MESSAGE(L"listctrl_1", BUTTOM_DOWN, OnDuiMsgListCtrl1Click)
    DUI_CONTROL_NAMEMSG_MESSAGE(L"listctrl_2", BUTTOM_DOWN, OnDuiMsgListCtrl2Click)
DUI_DECLARE_MESSAGE_END()
```

在编写自己的 Handler 事件处理类时候可以参考样例，定义消息映射宏，每个需要处理的消息定义一行内容，定义出哪个控件的什么消息需要处理，由哪个函数处理，消息处理函数必须按照固定的参数格式来编写，消息处理函数的样例如下：

```
// 显示信息对话框消息处理
LRESULT CDuiHandlerMain::OnDuiMsgMsgBoxButton1(UINT uID, CString strName, UINT Msg, WPARAM
wParam, LPARAM lParam)
{
    DuiSystem::DuiMessageBox(NULL, _T("演示对话框！"));
```

```
    return TRUE;
}
```

每个消息处理函数必须按照以上样例中的参数格式，函数的返回值表示此消息是否不再需要向下传递继续处理了，如果返回 TRUE，则消息处理结束，如果返回 FALSE，则此消息还可以继续被后面定义的其他函数处理。

使用以上的宏定义，会自动在进入消息处理函数时候记录日志，日志内容类似于下面这样：

DEBUG 2014-09-07 22:43:31[98544] : CDuiHandlerMain::OnDuiMessage:uID=1109, name=menu_1, msg=1, wParam=0, lParam=0

2.10. 控件的唯一标识和控件查找

DuiVision 库的一个特点就是可以通过 ID 和 name 两种方式灵活的进行控件的查找，每个控件对象创建的时候都会自动分配一个唯一 ID，同时也可以给控件命名，查找一个控件也可以通过 ID 和 name 两种方式进行查找，因为 ID 方式查找不够灵活，所以一般情况下都建议用 name 的方式进行查找，控件的 name 有两种方式可以设置，一种方式是在 xml 文件中定义控件的 name 属性，另一种方式是控件创建之后调用 SetName 函数进行设置。

控件查找可以通过 GetControl 函数进行查找，这个函数在事件处理基类、控件基类、对话框中都有定义。其中控件基类中 GetControl 函数的查找方式是递归查找当前控件的子控件，看是否有对应名字或 ID 的子控件；事件处理类中 GetControl 函数的查找方式是查找此事件处理对象关联的控件对象，然后查找此控件对象或子控件中是否有对应名字或 ID 的控件；对话框类中的 GetControl 函数的查找方式是递归查找当前对话框中的所有控件，看是否有对应名字或 ID 的控件。

2.11. 系统预定义控件、动作和事件

DuiVision 库中预定义了一些控件名、动作和事件，这些定义可以参考 duiid.h。

对于预定义的控件名，只要某个控件定义的名字是这个名字，就会被看做为特定的控件，系统会对其事件作出响应，预定义控件如下：

控件名	定义	说明
tray.icon	NAME_TRAY_ICON	托盘图标，系统托盘图标事件中的控件名都是这个名字
button.min	NAME_BT_MIN	最小化按钮，按钮定义成这个名字点击会最小化窗口
button.max	NAME_BT_MAX	最大化按钮，按钮定义成这个名字点击会最大化/恢复窗口

button.close	NAME_BT_CLOSE	关闭按钮，按钮定义成这个名字点击会关闭窗口
button.skin	NAME_BT_SKIN	换肤按钮，按钮定义成这个名字点击会显示换肤窗口
button.setup	NAME_BT_SETUP	设置按钮，建议打开系统设置窗口的按钮控件用这个名字
win.caption	NAME_AREA_CAPTION	窗口标题区域控件，在此控件位置范围内双击鼠标会发送非客户区鼠标双击事件，从而触发窗口的最大化或恢复，不论定义什么类型的控件都可以
frame.mainwnd	NAME_FRAME_MAINWND	主窗口的透明度渐变层蒙板图片，这个控件是每个窗口自动创建的
button.ok	NAME_BT_OK	确定按钮，用于对话框中的按钮，点击会指定对话框的 DoOK 函数
button.cancel	NAME_BT_CANCEL	取消按钮，用于对话框中的按钮，点击会指定对话框的 DoCancel 函数
button.yes	NAME_BT_YES	是按钮，用于对话框中的按钮，点击会指定对话框的 DoYes 函数
button.no	NAME_BT_NO	否按钮，用于对话框中的按钮，点击会指定对话框的 DoNo 函数
skin.wnd	NAME_SKIN_WND	皮肤选择窗口

系统预定义的动作如下，这些预定义动作用于定义在控件 xml 的 action 属性中：

action 名	定义	说明
close-window:	ACTION_CLOSE_WINDOW	关闭窗口，action 中以此开头，后面跟的是窗口名字，表示动作为关闭指定的窗口
hide-window	ACTION_HIDE_WINDOW	隐藏窗口，表示隐藏当前窗口
show-window:	ACTION_SHOW_WINDOW	显示窗口，action 中以此开头，后面跟的是窗口名字，表示动作为显示指定的窗口

预定义的系统消息：

消息定义	ID	说明

MSG_TRAY_DBCLICK	1	托盘双击消息
MSG_TRAY_LBUTTONDOWN	2	托盘左键单击消息

预定义的控件消息：

消息定义	ID	说明
MSG_BUTTON_DOWN	1	鼠标或键盘在控件按下
MSG_BUTTON_UP	2	鼠标或键盘在控件放开
MSG_BUTTON_DBLCLK	3	鼠标在控件双击
MSG_BUTTON_CHECK	4	检查框点击
MSG_SCROLL_CHANGE	5	滚动条位置变更事件
MSG_CONTROL_BUTTON	6	控件内的按钮点击事件 ,例如 tabctrl 控件中可以在每个 tab 页增加一个可点击的按钮，用于关闭 tab 等操作 ,这种控件内部按钮点击时候会发送此事件；edit 控件在编辑区或者 edit 中的小按钮点击时候，也会发送此事件
MSG_MOUSE_MOVE	7	鼠标在控件范围内移动的事件
MSG_MOUSE_LEAVE	8	鼠标离开控件范围的事件（离开之后只会发送一次此消息）
MSG_MOUSE_LDOWN	9	鼠标左键按下（仅用于鼠标事件，设置了 duimsg 属性之后才会发送）
MSG_MOUSE_LUP	10	鼠标左键放开（仅用于鼠标事件，设置了 duimsg 属性之后才会发送）
MSG_MOUSE_RDOWN	11	鼠标右键按下（仅用于鼠标事件，设置了 duimsg 属性之后才会发送）
MSG_MOUSE_RUP	12	鼠标右键放开（仅用于鼠标事件，设置了 duimsg 属性之后才会发送）
MSG_KEY_DOWN	13	控件的键盘按下事件处理，消息中的 wParam 表示键盘码 ,lParam 表示 Flasgs 状态(设置了 duimsg 属性之后才会发送）
MSG_CONTROL_EVENT	14	控件的自定义事件
MSG_MOUSE_RDBLCLK	15	鼠标右键双击
MSG_CONTROL_SELECT	16	控件内的选择事件 ,例如 combobox 控件的下拉选

		择框选择了一项之后，就会发送此事件
MSG_CONTROL_DELETE	17	控件内的删除事件 ,例如 combobox 控件的下拉选择框删除了一项之后，就会发送此事件
MSG_DROP_FILE	18	拖拽文件事件，可以用鼠标拖拽文件到设置了 dropfile 属性的控件，拖拽时候会发送此事件，其中 wParam 参数表示鼠标位置，lParam 参数表示文件的全路径名
MSG_FOCUS_CHANGE	19	控件的焦点状态变更时间，wParam 表示变更后的焦点状态，1 表示获取了焦点，0 表示取消了焦点
MSG_KEY_UP	20	控件的键盘放开事件处理，消息中的 wParam 表示键盘码 ,lParam 表示 Flasgs 状态(设置了 duimsg 属性之后才会发送)

2.12. 控件的快捷键和焦点的支持

每个控件可以设置快捷键，设置方法是在 xml 中设置 shortcut 属性，例如下面这个控件设置快捷键为 ESC 键：

```
<imgbtn name="button.close" pos="-45,0,-0,29" skin="IDB_BT_CLOSE"
shortcut="ESC" />
```

快捷键的写法是 flag+char 的形式，flag 可以是 CTRL、ALT、SHIFT，分别表示几个控制键是否按下，char 是键的名字，可以用的包括字母、数字，以及下面这些键：

RETURN – 回车

ESC – 取消

BACK – 回退

TAB – TAB 键

SPACE – 空格键

PRIOR – 上翻页

NEXT – 下翻页

END – 到文件尾

HOME – 到文件头

LEFT、UP、RIGHT、DOWN – 几个方向键

PRINT – 打印

INSERT – 插入

DELETE – 删除

F1-F12 – 功能键

如果没有 flag，也可以直接写键的名字。

某个控件如果设置了快捷键，当按下快捷键时候，相当于在控件上点击了一次鼠标，实际动作就是自动触发针对这个控件的一次鼠标按下消息和鼠标放开消息。

DuiVision 的控件支持焦点状态，如果一个控件要支持焦点的话，可以通过设置控件的 tabstop 属性来实现，tabstop 为 1 表示此控件可以处于焦点状态，tabstop 属性也可以通过 API 查询，就是控件的 IsTabStop 函数。一些控件默认是会设置为 tabstop 为 1 的状态的，这些控件包括按钮、检查框、RadioButton、编辑框。

焦点的切换有两种方式，一种是通过键盘操作，TAB 键和 SHIFT+TAB 键分别表示切换到下一个或上一个可以获取焦点的控件上面，另一种方式是鼠标点击了一个控件之后，这个控件就会成为当前的焦点控件。为了区分焦点控件，DuiVision 提供了默认的焦点控件显示方式，就是在控件的内部靠近控件边框位置画灰色的虚线框，有些控件不会采用这样的方式画焦点，例如编辑框处于焦点状态时候会显示编辑框的输入光标，而编辑框失去焦点时候会取消光标的显示，同时会在编辑框内显示出编辑框控件的 tip 信息，如果不想让控件在焦点状态时候显示虚线框，可以设置控件的 showfocus 属性为 0。

2.13. 控件的 tip 的支持

控件可以设置 tip 属性，也可以通过 API 函数 SetTooltip 来设置。

设置了 tip 之后，当鼠标移动到此控件并停留一段时间，就会显示此控件的 tip 信息，例如下面这个图所示：

2.14. 动态创建界面控件

有时候需要通过代码来动态创建控件，可以参考下面的动态创建控件的例子：

```
CDuiButton* pToolBtn =
static_cast<CDuiButton*>(DuiSystem::CreateControlByName(L"button",
m_pDlg->GetSafeHwnd(), m_pDuiObject));
if(pToolBtn)
```

```
{
    ((CControlBase*)m_pDuiObject)->AddControl(pToolBtn);
    pToolBtn->SetName(L"btnName");
    pToolBtn->SetTitle(L"动态按钮");
    pToolBtn->SetPosStr(L"10,10,110,35");
    pToolBtn->OnAttributeSkin(L"skin:IDB_BT_ICON", FALSE);
    pToolBtn->OnAttributeImageBtn(L"skin:IDB_ICON_TOOL", FALSE);
    pToolBtn->SetMaxIndex(4);
    pToolBtn->SetAction(L"dlg:dlg_login");
    pToolBtn->OnPositionChange();
}
```

这段代码首先通过 DuiSystem 提供的动态创建控件的函数（CreateControlByName）来创建一个控件（创建时候需要指定控件类型名，例如按钮控件就是 button、所在的窗口的句柄、父控件指针）。

创建之后再调用控件的函数来设置控件的一些属性，需要特别注意的是控件的位置信息的设置，必须使用位置字符串的方式来设置（对应控件的 pos 属性），因为位置字符串是可以根据父控件的位置信息计算出控件的实际位置的，如果不采用这种方式，直接指定控件的位置，则可能会因为位置自动计算时候没有相关的信息导致最后计算不出真正的位置。

位置字符串设置需要调用 SetPosStr 函数进行设置，注意最后一定要调用 OnPositionChange 函数，这个函数会真正进行一次位置计算，这样才能使控件显示在正确的位置。

2.15. 用户自定义控件

如果 DuiVision 库中的控件不能满足需求，也可以开发自己的自定义控件，自定义控件可以从 CcontrolBase 控件基类或其他的某个控件类派生，具体开发方法可以参考 DuiVision 库中的控件代码。

自定义的控件代码开发完成后，将代码添加到用户自己的代码工程中，然后在控件使用之前注册到 DuiVision 库中就可以，建议注册代码放在主程序的 DuiSystem 初始化之后，下面的代码演示了在主程序中注册类名为 CDuiUserControl 的自定义控件的方法，蓝色部分代码就是用于注册自定义控件的代码：

```
BOOL CDuiVision1App::InitInstance()
{
    CWinApp::InitInstance();

    AfxEnableControlContainer();

    // TODO: 应适当修改该字符串，
```

```
    // 例如修改为公司或组织名
    SetRegistryKey(_T("DuiVision1"));

    // 初始化DuiVision界面库,可以指定语言,dwLangID为表示自动判断当前语言
    // 11160是应用程序ID,每个DUI应用程序应该使用不同的ID,ID主要用于进程间通信传递命令行时候区分应用
    DWORD dwLangID = 0;
    new DuiSystem(m_hInstance, dwLangID, _T("DuiVision1.ui"), 11160, IDD_DUIVISIONAPP_DIALOG, _T(""));

    // 注册用户自定义控件控件
    REGISTER_DUICONTROL(CDuiUserControl, NULL);

    // 创建主窗口
    CDlgBase* pMainDlg = DuiSystem::CreateDuiDialog(_T("dlg_main"), NULL, _T(""), TRUE);
    // 给主窗口注册事件处理对象
    CDuiHandlerMain* pHandler = new CDuiHandlerMain();
    pHandler->SetDialog(pMainDlg);
    DuiSystem::RegisterHandler(pMainDlg, pHandler);

    // 初始化提示信息窗口
    DuiSystem::Instance()->CreateNotifyMsgBox(_T("dlg_notifymsg"));

    // 按照非模式对话框创建主窗口,可以默认隐藏
    pMainDlg->Create(pMainDlg->GetIDTemplate(), NULL);
    INT_PTR nResponse = pMainDlg->RunModalLoop();

    // 释放DuiVision界面库的资源
    DuiSystem::Release();

    // 由于对话框已关闭，所以将返回FALSE 以便退出应用程序，
    //  而不是启动应用程序的消息泵。
    return FALSE;
}
```

2.16. 日志文件定义

DuiVision 提供了日志文件的操作函数，在任何地方都可以通过如下代码调用日志函数来写日志：

```
    DuiSystem::LogEvent(LOG_LEVEL_DEBUG, _T("CDuiHandlerTab3::OnDuiMessage:uID=%d, name=%s, msg=%d, wParam=%d, lParam=%d"),
```

uID, strName, Msg, wParam, lParam);

此函数的第一个参数是日志的级别，后面的参数类似于 C 语言的 printf 的写法。级别定义如下：

```
#define LOG_LEVEL_DEBUG   0x0001        //调试信息
#define LOG_LEVEL_INFO    0x0002        //一般信息
#define LOG_LEVEL_ERROR   0x0004        //错误信息
#define LOG_LEVEL_CRITICAL 0x0008        //致命信息
```

日志文件默认是在 exe 所在路径下面，日志文件名和日志级别可以在全局资源定义（resource.xml）中定义，定义如下：

```
<!--系统配置-->
<res type="cfg" name="logfile" value="demoui.log" />
<res type="cfg" name="loglevel" value="1" />
<res type="cfg" name="logFileSize" value="1024" />
<res type="cfg" name="logFileNumber" value="5" />
```

其中 logfile 表示日志文件的名字，是对应的 exe 文件的路径，loglevel 表示记录的日志的最低级别，日志级别有 4 种级别，分别是 1-调试级别、2-一般级别、4-错误级别、8-致命级别。如果这里的 loglevel 定义的是 1，就表示调试和以上级别的日志都会记录。

logFileSize 表示单个日志文件的大小上限，超过这个指定的大小，则会将旧的日志文件备份之后创建新的日志文件，这个大小的默认单位是 K 字节。

logFileNumber 表示备份的日志文件的最大个数，最多只会保留这么多的日志文件。

2.17. 任务类

基于 MFC 的界面程序中，如果存在多线程，一般情况下只有主线程（界面线程）可以调用 Windows 窗口相关的函数，否则如果在其他线程中调用了界面函数，很可能会造成异常。为此 DuiVision 库提供了一个任务队列和相应的调度机制，可以将各种任务对象放到任务队列中按顺序执行，通过任务队列，可以做到其他线程和界面线程之间的中转调用，方法是创建任务对象时候指定是需要界面线程处理的任务，则任务调用过程中会通过跨进程的消息将此任务通知界面线程来执行。

DuiVision 中定义了任务的基类和几种派生类，基类是 IBaseTask 类，派生类目前有两种，分别是 DuiSystem 中定义的 CDuiActionTask 类（用于执行菜单等动作），另一种是 DuiSystem 中定义的 CDuiNotifyMsgTask 类，用于执行桌面右下角的弹出提示窗口，这两个任务类的核心执行代码都在 DoAction 函数中，如果需要派生其他的任务类，可以参考者两个类的实现方式。

2.18. 定时器

DuiVision 库提供了定时器功能的封装，定时器的封装类是 CTimer 类，CDlgBase、CDlgPopup 都会从 CTimer 继承，实现定时器的功能。

CTimer 中封装了两种类型的定时器实现，一种是多媒体定时器，另一种是 windows 定时器。多媒体定时器的精度比较高，但存在的问题是如果定义了周期触发的定时器，程序运行过程中操作系统休眠了，再恢复之后，定时器中间间隔的那些执行周期都会被依次触发，这就会造成像动画定时器如果没有停止，休眠恢复之后一段时间内 CPU 占用率会很高，因为要把休眠阶段的定时周期都执行完，需要不停的刷新界面，导致 CPU 占用高；windows 定时器没有这个问题，但目前 windows 定时器和 activex 控件有冲突，如果程序中使用了 activex 控件会出现显示异常。

CTimer 中的两种定时器实现可以通过编译宏来决定使用哪种定时器，默认是使用的多媒体定时器，编译宏的定义代码在 timer.h 头文件中，如下所示：

```
//#define USE_WND_TIMER  // 使用windows定时器
#define USE_WMM_TIMER   // 使用多媒体定时器
```

DuiVision 库中默认会使用的定时器包括动画定时器和窗口自动关闭定时器，动画定时器由 DlgBase 或 DlgPopup 中窗口创建的时候就会创建，每个窗口只有一个动画定时器，定时周期是 30 毫秒，DlgBase 或 DlgPopup 的动画定时器会在每个定时周期调用所有子控件的 OnTimer 函数，控件如果需要显示动画只要在 OnTimer 函数中处理就可以了，动画周期都是 30 毫秒的整数倍。窗口自动关闭定时器用于设置了自动关闭的窗口，在关闭时间到了之后会触发，从而调用窗口的关闭函数。

应用程序中也可以创建自己的定时器，DuiSystem 已经封装了定时器相关的函数，主要包括如下几个函数：

```
static UINT AddDuiTimer(UINT uTimerResolution, CString strTimerName = _T(""), BOOL
bAppTimer = TRUE);
static void RemoveDuiTimer(UINT uTimerID);
static void RemoveDuiTimer(CString strTimerName);
```

可以创建和删除自定义的定时器，创建时候可以指定定时周期、定时器名字、是否应用的定时器，如果指定了是应用定时器，则定时周期到了之后会调用事件处理类的 OnTimer 函数，并且把定义的定时器名字传进去，在事件处理类的 OnTimer 函数中可以根据定时器 ID 和名字对定时器进行判断。

2.19. 进程间通信

DuiVision 可以通过配置 appMutex xml 属性使应用程序实现只能同时运行一个应用程序实

例，无论是否设置了这个属性，都可能会有同一个应用程序的多个进程实例之间需要通信的需求。对于只能运行一个应用实例的场景，实现方式是启动第二个进程实例时候会判断是否注册了指定的 mutex，如果已经注册了，则第二是实例会立即退出，但是退出之前可能会需要发送消息给第一个实例，让第一个实例把界面显示出来；对于不限制运行多个实例的场景，也可能会有需求传递一些消息给其他的实例。

为了简化进程间通信，DuiVision 将进程间通信做了封装，整体的实现原理是：

1、DuiSystem 封装了一个发送进程间消息的 SendInterprocessMessage 函数，调用时候可以指定消息 ID、参数、应用名字、其他的字符串信息，一般应用名字可以用 xml 中定义的 appMutex 属性，这个函数会将传入的参数封装成一个 DUI_INTERPROCESS_MSG 结构体，然后放在一个内存映射文件中（文件名固定），同时发送一个跨进程的广播消息，消息 ID 是 WM_CHECK_ITS_ME，在确保消息发送完成之后，内存映射文件才会删除；

2、DlgBase 中有一个针对 WM_CHECK_ITS_ME 消息的处理函数，接收到此消息之后会根据其中一个参数判断是否当前应用进程需要处理的（判断是否和当前应用的 AppID 一致），然后打开内容映射文件获取到 DUI_INTERPROCESS_MSG 结构体，判断其中的应用名是否一致，如果一致则给当前对话框指定的 Handler 事件处理对象发送一个 DUI 消息，消息中的控件 ID 是 APP_IPC，是一个预定义的控件 ID，表示进程间通信用的；

3、只要在主对话框关联的 Handler 事件处理类定义 APP_IPC 控件对应的事件处理函数，就可以对进程间通信消息进行处理了，处理方法和普通的 DUI 消息是类似的，可以参考系统默认的 OnDuiMsgInterprocess 处理函数，其中对命令行参数和激活程序的窗口做了处理。

2.20. 控件的颜色属性说明

DuiVision 的很多控件中会有颜色类型的属性，如果没有特殊说明，这些颜色类型的属性在 xml 定义中支持三种格式，分别是如下的格式：

1、16 进制表示的格式 rrggbb，例如 crtext="8090A0"，表示颜色值 red=0x80，green=0x90，blue=0xA0；

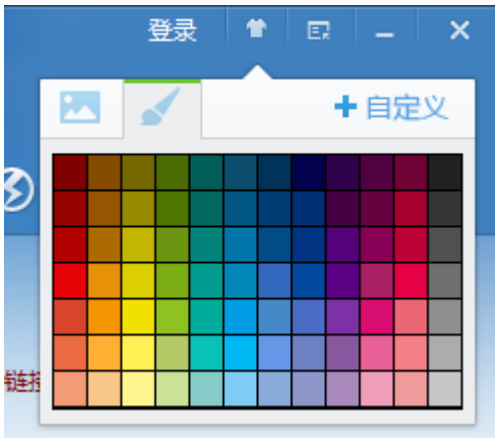
2、十进制的 argb 格式，ARGB 是一种色彩模式，也就是 RGB 色彩模式附加上 Alpha（透明度）通道，例如 crback="255,128,224,128"，表示 alpha=255，red=128，green=224，blue=128，其中 alpha=255 表示不透明，如果为 0 表示全透明；

3、十进制的 rgb 格式，例如 crtext="128,224,128"，表示 red=128，green=224，blue=128。

其中第二和第三种格式的判断方法是看有几个逗号分隔符，如果有 3 个逗号分隔符，就是 argb 格式，否则就是 rgb 格式。

2.21. 界面皮肤定义

DuiVision 界面库支持定义窗口的背景皮肤，如果某个窗口未指定特殊的背景皮肤，则会使用全局的背景皮肤，通过界面库提供的皮肤窗口可以更改界面皮肤，皮肤窗口如下：



通过皮肤窗口可以将背景皮肤更换为默认的 9 张图片之一，或者是选择的某种颜色，或者选择一个自定义的图片背景。

皮肤选择窗口的定义文件是 dlg_skin.xml，如果想修改皮肤窗口的界面，可以修改这个文件，默认的 9 张背景图片是放在 exe 所在路径的 bkimg 子目录下，图片资源名是 SKIN_PIC_0-SKIN_PIC_8，图片文件名分别是 SKIN_PIC_0.png-SKIN_PIC_8.png，如果要更改默认的图片，可以在制作安装包时候替换这个目录中相应名字的图片，或者修改 resource.xml 中 SKIN_PIC_0-SKIN_PIC_8 定义的文件。

皮肤选择窗口是通过皮肤按钮来打开的，只要窗口的某个控件定义的名字是皮肤按钮的名字，就自动具有皮肤按钮的功能，皮肤按钮的名字是 button.skin，例如下面这段对话框 xml 中的定义，就自动会支持皮肤窗口的功能：

```
<imgbtn name="button.skin" pos="-140,0,-111,29" skin="IDB_BT_SKIN"
tip="皮肤" show="1"/>
```

通过皮肤窗口更改了背景皮肤之后，这个改动并不会保存下来，如果下次运行这个程序，就

会又回到最初的状态（默认显示的是第一张背景图片），为了将选择的皮肤保存下来，需要在主程序中写一些代码来实现，例如将选择的皮肤信息保存在注册表中，具体实现方式可以参考 DuiVision 界面库 demo 程序中的 DuiHandlerMain.cpp 的 CDuiHandlerMain::OnDuiMsgSkin 函数的实现。

2.22. 托盘图标

DuiVision 界面库封装了 Windows 托盘图标的相关操作，可以创建托盘图标，并设置图标文件、托盘的 tip 信息，也可以处理托盘的单击、双击、右键菜单的事件。

通过调用下面的函数可以进行托盘的初始化：

```
DuiSystem::Instance()->InitTray();
```

初始化一般放在主的事件处理类 OnInit 函数中，可以参考 demo 程序的代码。设置托盘的图标文件盒 tip 信息可以调用 DuiSystem 的 SetTrayIcon、SetTrayTip 函数。

托盘的右键操作是打开右键菜单，右键菜单在 resource.xml 中通过 menu_tray 名字的资源项定义具体的菜单 xml 文件。

托盘的左键双击默认动作是打开主窗口，也可以更改为自定义的处理方式，resource.xml 中下面的配置项用于定义托盘双击的动作，如果为 0 就表示执行默认的打开主窗口的动作，如果为 1，则会发送 MSG_TRAY_DBCLICK 消息，通过在事件处理类中响应这个消息，就可以处理双击事件。

```
<res type="cfg" name="trayDbClickMsg" value="0" />
```

托盘左键的单击事件也会发送一个消息，消息 ID 为 MSG_TRAY_LBUTTONDOWN，通过在事件处理类中响应这个消息，就可以处理单击事件。可以参考 Demo 程序单击和双击事件响应函数。

2.23. DPI 虚拟化设置

DPI 全称是 dots per inch (DPI)， 也就是每英寸的点数，在显示器上就是每英寸的像素个数，Window 上一般默认是 96 dpi 作为 100% 的缩放比率， 但是要注意的是该值未必是真正的显示器物理值， 只是 Windows 里我们的一个参考标准。

随着显示器分辨率的提高，如果还用传统的 96dpi，就会导致应用程序在界面中显示的太小，看不清楚，因此就出现了高 DPI 的支持问题。

关于高 DPI 的支持， Windows XP 时代就开始有了，那时关于高 DPI 的支持比较简单， 但是从 Vista/Win7 到现在 Win8 /Win8.1， Windows 关于高 DPI 的支持已经发生了很大的变化。通常来说如果我们程序支持高 DPI， 意味着我们要对绘画的内容进行相应的放大， 比如字体，图片和控件等。当然， 如果我们用的是系统字体（比如

GetStockObject(DEFAULT_GUI_FONT))，那么这种情况下我们不用操心， 因为系统会对该字体在高 DPI 时进行相应的放大； 如果我们是用 CreateFont 自己创建的字体， 那就要我们自己对该字体进行放大了。

Windows XP 对高 DPI 的支持比较差劲， 大部分情况下就是字体的放大， 当然我们程序也可以通过 GetDeviceCaps(hDC, LOGPIXELSX)获取 DPI 后自己对绘画的内容进行缩放。

针对那些根本不关 DPI 的软件，Windows Vista 引入了一项新技术“DPI Virtualization”(DPI 虚拟化)。如果 DPI 不超过 120，继续使用 XP 的缩放机制，超过之后就开启这个新模式。

其实说起来也很简单，所谓的 DPI 虚拟化，只不过是系统欺骗那些不支持高 DPI 的软件，让它们以为 DPI 其实是最基本的 96(相当于 1280×1024 分辨率的 17 寸显示器)，然后由系统离屏渲染软件界面，再放大到用户选择的 DPI 级别上。这样解决了文字乱跑的问题，但很容易造成显示模糊，并不是完美的解决方法

为此，微软引入了另外一个新 API SetProcessDpiAwareness，本身支持 DPI 缩放的软件可以用它告诉系统不要对这个进程进行 DPI 虚拟化。

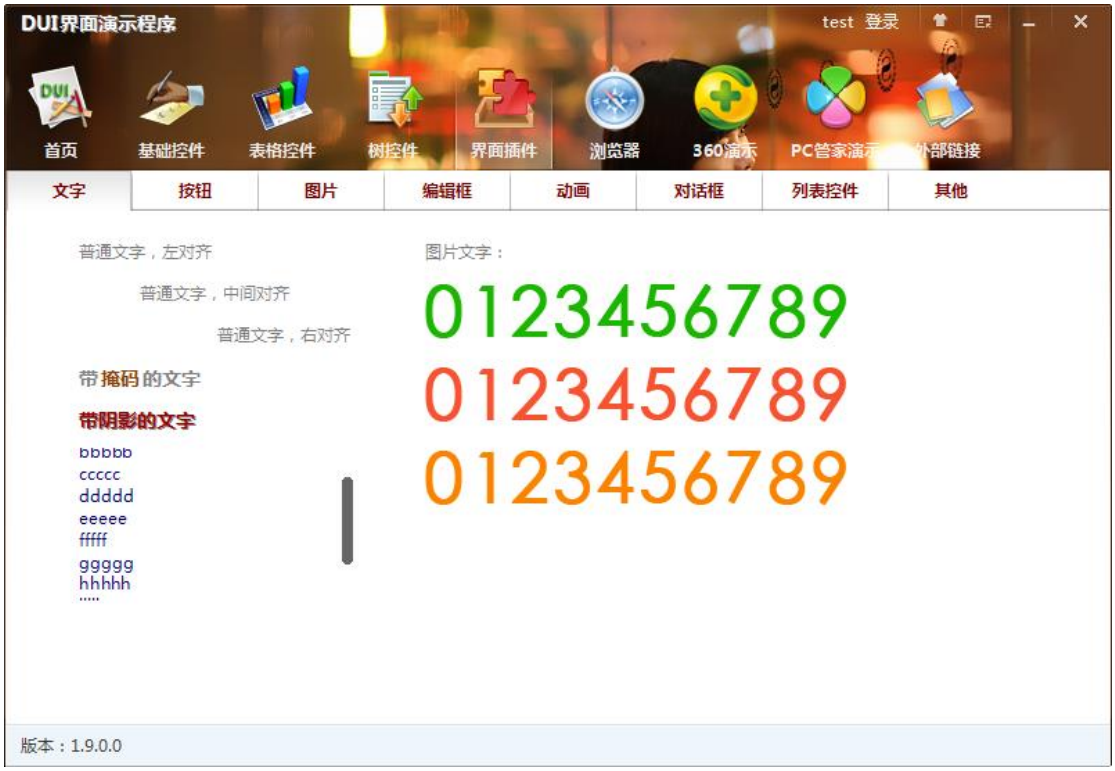
DuiVision 本身并没有做高 DPI 的相关处理，但使用 DuiVision 开发的应用程序如果想自己做高 DPI 的处理，而不是使用系统默认的 DPI 虚拟化，可以在 resource.xml 中增加下面这个设置，也就是 dpiAware 设置为 1，表示禁用 DPI 虚拟化，默认是 0，表示不禁用。

```
<res type="cfg" name="dpiAware" value="1" />
```

2.24. 界面插件介绍

DuiVision 界面库支持界面插件，每个界面插件是一个独立的动态库，里面可以封装各种界面，界面插件会对外暴露标准的插件接口，主应用程序可以加载插件，将插件中界面集成到主程序指定的位置，插件中可以有自己的事件处理。通过插件可以将一个大的界面应用程序分离成多个小程序，DuiVision 界面插件的一个优点是基于接口进行集成，所以主程序和界面插件动态库可以使用不同版本的 DuiVision 库进行编译，减少了主程序和插件的版本依赖性，甚至基于其他界面库开发的界面，只要是符合界面插件规范也可以进行集成。

DuiVision 界面插件目前只支持通过 div 控件进行集成，因为 div 控件是一个容器控件，通过设置 div 控件的 plugin 和 plugin-debug 属性，指定插件的动态库文件，主程序在初始化这个 div 时候就会自动加载插件动态库，并将插件中的顶层 div 和主程序中这个 div 关联起来，插件显示的区域就是主程序这个 div 指定的区域。插件显示的效果如下图，下图中当前的这个界面插件 tab 页中显示的内容都是通过插件显示出来的。



通过 DuiVision 工程向导可以创建出插件工程的框架代码，具体用法也可以参考 Demo 程序中的插件演示页面。

2.25. 界面演示

可以参考 demo 程序，demo 程序的一些界面如下：


DUI界面演示程序
test 登录
首页
基础控件
列表控件
表格控件
树控件
浏览器
360演示
PC管家演示
外部链接
DUI界面库介绍
DuiVision界面库是参考了仿PC管家程序、金山界面库、DuiEngine、DuiLib等多个基于DirectUI的界面库开发的。
DirectUI技术一般是指将所有的界面控件都绘制在一个窗口上，这些控件的逻辑和绘图方式都必须自己进行编写和封装，而不是使用Windows控件，所以这些控件都是无句柄的。
DirectUI技术需要解决的主要问题如下：
1、窗口的子类化，截获窗口的消息；
2、封装自己的控件，并将自己的控件绘制到该窗口上；
3、封装窗口的消息，并分发到自己的控件上，让自己的控件根据消息进行相应和绘制；
4、根据不同的行为发送自定义消息给窗口，以便程序进行调用；
5、一般窗口上控件的组织使用XML来描述。
通常DirectUI的界面库都采用XML配置文件+图片+控制脚本（Lua、Javascript等）的开发方式，非常类似于Web程序的开发方式，当然这里面控制脚本也可以直接使用C++代码来实现。这种开发方式可以大大提高开发效率，将程序员从繁琐的界面工作中解脱出来，并且通过美工的设计，可以使界面更美观。
DuiVision开源下载地址：https://github.com/blueantst/DuiVision
蓝蚂蚁工作室网站：http://www.blueantstudio.net
QQ群：325880743
微信公众号：blueantstudio 或搜索 蓝蚂蚁工作室 或扫描左侧二维码
版本：1.0.0.1


DUI界面演示程序-仿360安全卫士
登录
电脑体检
木马查杀
系统修复
电脑清理
优化加速
软件管家
您的电脑已经 5 天没有体检，建议立即体检
系统可能已经存在大量风险，安全度和性能都在急速下降
建议您每天坚持电脑体检，提高电脑的安全和性能
立即体检
版本：1.0.1.1


DUI界面演示程序
test 登录
首页
基础控件
表格控件
树控件
界面插件
浏览器
360演示
PC管家演示
外部链接
表格控件
Script.NET
蓝蚂蚁工作室开发的脚本语言集成开发环境，支持Tcl、...
已装：2.2.1.0
最新：4.0开源版
125.5M
点击下载
腾讯软件管理
更新内容
1.新的界面，丰富的内容；2.海量软件搜索；3.功能分类清...
已装：8.7.0.1
最新：10.0.0.1
16.5M
一键安装
Notepad++
1.修复功能列表崩溃问题;2.增强分隔符选择
已装：6.5.0.0
最新：6.6.0.0
7.5M
一键安装
迅雷看看
1.修复部分 Bug
已装：4.8.0.0
最新：4.9.14.2
43.5M
一键安装
Axialis IconWorkshop
1.易用性改进
已装：4.9.0.0
最新：6.8.1.2
13M
一键安装
为知笔记
1.改进了博客发布功能;2.改进了编写日记功能
已装：22.9.0.0
最新：24.0.1.1
29.5M
TortoiseGit 64位
暂无版本更新说明
已装：1.8.3.0
最新：1.8.9.0
7.5M
Notepad++
版本：1.9.0.0

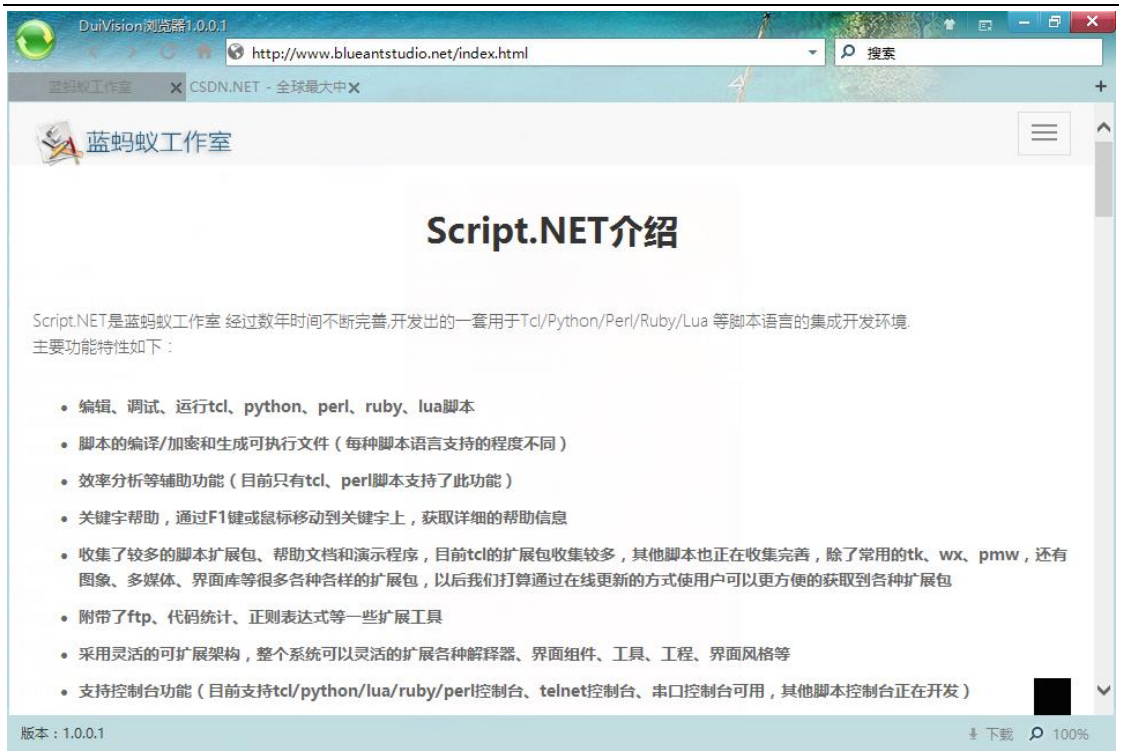

DUI界面演示程序
test 登录
首页
基础控件
列表控件
表格控件
树控件
浏览器
360演示
PC管家演示
外部链接
蓝蚂蚁工作室
http://www.blueantstudio.net
主页
下载中心
软件介绍
插件
关于
英文版
Search:
软件作品
Script.NET开发平台
Script.NET开源项目
FVision界面类库
金山界面库VC向导
下载中心
Script.NET
FVision
金山界面库VC向导
LuaEditor
blueant
Script.NET V2.2 版本发布
Tcl
Python
Perl
Ruby
Lua
Tcl 脚本开发支持
Python 脚本开发支持
Perl 脚本开发支持
Ruby 脚本开发支持
Lua 脚本开发支持
.....
Script.NET 通用开发平台支持多种
脚本语言的开发、调试、运行 ...
立即下载
版本：1.0.0.1

以上界面中的整体布局和实现方式是，在主窗口 xml 中定义了一个 tabctrl 控件，控件中定义了几个 tab 页，每个 tab 页下面定义具体这个页面的控件，例如第三个页面中定义了一个 listctrl 列表控件和两个图片按钮控件。

用户点击了某个页面上的某个控件之后，需要做一些相应的处理，这些处理需要写在事件处理类中，demo 程序只定义了一个事件处理类，可以处理所有与页面的事件，也可以给某个 tab 页单独指定事件处理类，只要将处理某个页面的事件处理对象注册到这个页面控件就可以，对于其他没有注册事件处理对象的页面中的事件，会自动遍历每个注册的事件处理对象进行处理。具体实现请参考 demo 程序代码。

下面是 DuiVision IM 演示程序的截图：

下面是 DuiVision Explorer 演示程序的截图：



3. 控件说明

3.1. DUI 基类

类名：CDuiObject

控件名：无（基类，没有实体控件）

说明：所有 DUI 的基类（包括控件、对话框、菜单等）

属性：无

函数：

函数	是否虚函数	说明
IsClass	是	判断是否此类
GetObjectClass	是	获取类名
BaseObjectClassName	是	获取基类名
GetID	否	获取对象 ID
GetName	否	获取对象名字
IsThisObject	否	根据 ID 或名字判断是否此对象，ID 或名字任意一个匹配上就可以
RegisterHandler	否	注册事件处理对象
GetDuiHandler	否	获取事件处理对象指针
GetRect	否	获取控件的左上角、右下角坐标
ParseDuiString	否	解析字符串，替换其中的替换内容（用[]包围的定义内容），替换内容是在 resource.xml 或字符串定义文件中定义的字符串内容
ParseKeyCode	否	根据传入的字符串获取对应的键盘码，例如 CTRL+F1 会被转换为 0x11,0x70

3.2. DUI 控件基础类

类名：CControlBase

控件名：无（基类，没有实体控件）

说明：所有界面控件的基类

属性：

属性名	类型	说明
show	1|0	控件是否可见
disable	1|0	控件是否被禁用
pos	位置	控件的位置坐标，可以是左上角坐标，例如 10,10，也可以是左上角+右下角坐标，例如

		10,10,200,100。 支持正值和负值和百分数，正值表示从父控件左上角开始计算的值，负值表示从父控件右下角开始计算的值，百分数表示按照父控件宽度和高度的百分比，例如-10,10 表示从右边往左 10 像素，从上往下 10 像素的位置，%50,%50 表示父控件中间的位置。 也可以支持从父控件中间开始计算的坐标，用|表示中间位置 ,例如|10,|-10 表示横向中间向右 10 像素，纵向中间向上 10 像素的位置。
width	数字	控件宽度
height	数字	控件高度
action	字符串	控件的动作字符串 ,表示控件点击之后执行的动作，有几种类型： 1、dlg:xxxx，表示显示指定的对话框，显示的对话框可以是一个 xml 定义文件 ,也可以是一个定义(从 resource 资源定义文件中查找具体的 xml 定义文件) 2、menu:xxxx.xml，表示显示指定的菜单，菜单定义文件是指定的 xml 文件 3、link:url，表示使用浏览器或其他默认程序打开指定的链接或文件 4、run:xxx.exe|param，表示运行指定的程序，可以传递命令行，执行的程序后面用|分隔，|之后的表示传递的命令行参数
menupos	位置	菜单显示的位置坐标，必须是 x,y,dx,dy 的形式（ dx,dy 为可选参数)。 **x,y** 表示菜单左上角坐标，例如 10,10。 支持正值和负值和百分数，正值表示从父控件左上角开始计算的值，负值表示从父控件右下角开始计算的值，百分数表示按照父控件宽度和高度的百分比，例如-10,10 表示从右边往左 10 像素，从上往

		下 10 像素的位置，%50,%50 表示父控件中间的位置。 也可以支持从控件中间开始计算的坐标，用|表示中间位置，例如|10,|-10 表示横向中间向右 10 像素，纵向中间向上 10 像素的位置。 **dx,dy** 是可选内容，表示菜单的显示方向，默认是正值，如果 dx 是负值，表示向左侧的显示方向，如果 dy 是负值，表示向上的显示方向。
tip	字符串	定义鼠标移动到空间上过一段时间会出现的 tip 提示信息，tip 信息只有对话框的基础控件可以定义，其他控件即使定义了也没有效果
tip-width	数字	Tooltip 的宽度，默认为 0，这个值只有设置为大于 0 才会有效
response	1|0	控件是否可以响应鼠标事件，如果不希望控件响应鼠标事件，可以设置此属性为 0
tabstop	1|0	控件是否可以响应焦点切换的事件，也就是按键盘 tab 键能否将焦点切换到此控件
taskmsg	1|0	调用事件响应函数时候是否采用任务方式，对于可能产生阻塞或耗时比较长的处理操作应该采用任务方式处理，避免造成界面不响应或界面异常
img-ecm	1|0	是否使用图片自身的颜色管理信息，默认为 0，表示加载图片文件时候使用系统的颜色管理信息而不使用图片自身的颜色管理信息，因为 XP SP1 以前的操作系统自带的 GDI+模块可能不支持图片自身的颜色管理信息，因此如果设置为 1 的话，在 XP SP1 以前的系统下运行图片可能无法正常显示
drag	1|0	是否允许鼠标拖动控件的位置，设置为 1 则允许拖动，默认是 0
dropfile	1|0	是否允许鼠标拖拽文件到此控件，如果设置为 1，则允许拖拽文件，拖拽发生时候会发送 MSG_DROP_FILE 消息
shortcut	字符串	定义控件的快捷键，快捷键字符串参考第二章的相

		应章节说明，当按下对应快捷键的时候，会自动触发此控件产生一个鼠标按下和鼠标放开的事件，模拟点击了此控件
cursor	字符串	定义控件的鼠标光标，如果定义，则鼠标移动到控件范围内会显示指定的鼠标光标，目前支持系统预定义的几个鼠标光标，分别是： arrow - 箭头；wait – 沙漏等待；cross – 十字；sizewe – 双箭头指向东西；sizens – 双箭头指向南北；hand – 手型；help – 箭头+问号
duimsg	字符串	指定控件会发送哪些 DUI 消息，消息名之间用|分隔，例如：” msg1|msg2|msg3” 。 目前支持的消息名包括： mousemove – 发送鼠标移动和鼠标离开控件的消息 mouseldown - 发送鼠标左键按下的消息 mouselup - 发送鼠标左键放开的消息 mouseldblclk - 发送鼠标左键双击的消息 mouserdown - 发送鼠标右键按下的消息 mouserup - 发送鼠标右键放开的消息 mouserdblclk - 发送鼠标右键双击的消息 keydown – 发送键盘按下的消息 keyup – 发送键盘放开的消息 focuschange – 发送控件焦点状态变更消息

函数：

函数	是否虚函数	说明
GetParent	否	获取父控件对象指针
GetControl	否	根据 ID 或 name 获取控件对象指针
AddControl	否	添加控件
RemoveControl	否	删除控件
GetControls	否	获取所有控件对象的列表
GetParentDialog	否	获取控件所在的对话框的指针

OnMessage	是	控件的消息处理函数
SendMessage	否	发送通知消息
SetRect	是	设置控件的位置
GetRect	是	获取控件的位置
OnPositionChange	是	刷新控件的位置信息
SetPosStr	否	设置控件的位置字符串
SetVisible	否	设置控件的可见性
GetVisible	否	获取控件的可见性
SetControlWndVisible	是	设置控件中的 Windows 原生控件是否可见的状态，由 Panel 对象中调用，对于 edit、activex 等使用了 Windows 原生控件的类需要重载此函数，并正确的进行原生控件的显示和隐藏
SetDisable	否	设置控件是否禁用
GetDisable	否	获取控件是否禁用
SetTabStop	否	设置控件是否能响应 tab 键
IsTabStop	否	获取控件是否能响应 tab 键
SetTooltip	否	设置控件的 tip 信息
GetTooltip	否	获取控件的 tip 信息
SetAction	否	设置控件的动作字符串
GetAction	否	获取控件的动作字符串
SetRresponse	否	获取控件是否可响应鼠标事件
GetRresponse	否	设置控件是否可响应鼠标事件
GetDragEnable	否	获取控件是否可以通过鼠标拖动位置
SetDragEnable	否	设置控件是否可以通过鼠标拖动位置
PtInRect	是	判断坐标是否在控件范围内
UpdateControl	否	刷新控件显示
SetWindowFocus	是	设置窗口焦点
OnFocus	否	设置此控件是否为当前焦点控件
IsFocusControl	否	判断此控件是否为当前焦点控件
OpenDlgPopup	否	打开弹出对话框
CloseDlgPopup	否	关闭弹出对话框

3.3. DUI 文字控件基础类

类名：CControlBaseFont

控件名：无（基类，没有实体控件）

说明：所有界面控件中有显示文字信息的控件类的基类

属性：

属性名	类型	说明
title	字符串	控件的显示标题
font	字体	控件的字体，可以引用资源定义中定义的某个字体，默认字体是 default
fontname	字符串	直接指定某种字体
fontwidth	数字	直接指定字体宽度
height	数字	控件高度
valign	枚举	文字的垂直对齐模式，top、middle、bottom
align	枚举	文字的水平对齐模式，left、center、right
skin	皮肤	控件的皮肤名，引用资源定义中的统一皮肤定义
image	图片	控件的图片，有 3 种定义方式： 1、图片文件：xxx.png，xxx.jpg 等，是相对 exe 的路径 2、图片资源：如果 image 不是文件格式，则认为是资源 ID，到程序的内嵌资源中去查找对应的图片资源 3、皮肤方式：skin:xxxx，如果是 skin:开始，则认为是皮肤格式，后面是皮肤名，到全局皮肤定义中查找具体图片
img-count	数字	定义图片的切片个数，如果一个图片文件中横向包含了多个等宽的小图片，根据这个定义，控件可以知道到底有几个小图片，并按照图片个数进行正确的切片

函数：

函数	是否虚函数	说明
SetTitle	否	设置标题文字

GetTitle	是	获取标题文字
SetAlignment	是	设置控件的水平对齐方式
SetVAlignment	是	设置控件的垂直对齐方式
SetImage	否	设置控件的图片，一般一个图片是由水平方向切分的多个大小相同的小图片组成的，小图片按顺序分别表示正常正太、鼠标移动状态、鼠标按下状态、禁用状态对应的图片，每种控件对应的小图片个数可能会有差别
SetBitmapCount	否	设置控件图片的水平方向切分的小图片个数

3.4. DUI 矩形区域控件

类名：CArea

控件名：area

说明：可以设置区域的渐变透明度，不能响应鼠标事件，此控件的原理是画一个填充的透明度渐变矩形区域，透明度从矩形区域顶部到底部均匀渐变。

属性：

属性名	类型	说明
color	颜色	矩形区域的颜色
transparent-type	枚举	vertical – 垂直方向渐变 horizontal – 水平方向渐变
begin-transparent	字体	矩形区域顶部的起始透明度
end-transparent	字符串	矩形区域底部的终止透明度

3.5. DUI 矩形控件

类名：CRectangle

控件名：rect

说明：可以画矩形边框，可以指定边框颜色、宽度、类型。

属性：

属性名	类型	说明
color	颜色	矩形区域的填充颜色
crframe	颜色	边框颜色
dashstyle	枚举	边框类型，支持的类型包括：

		solid – 实线 dash –由划线段组成的直线 dot –由点构成的直线 dashdot –由重复的划线点图案构成的直线 dotdot - 由重复的划线点点图案构成的直线 custom - 用户定义的自定义划线段样式
Frame-width	数字	边框宽度
transparent-type	枚举	vertical – 垂直方向渐变 horizontal – 水平方向渐变
begin-transparent	字体	矩形区域顶部的起始透明度
end-transparent	字符串	矩形区域底部的终止透明度

界面示例：

3.6. DUI 对话框

类名：CDlgBase

控件名：无

说明：对话框类，所有对话框都直接创建一个 CDlgBase 对象就可以。

代码中如果需要创建一个对话框，一般建议使用 DuiSystem 类中封装的若干对话框相关的函数来操作，包括创建对话框、删除对话框、根据对话框名获取对话框指针、显示通用对话框。

属性：

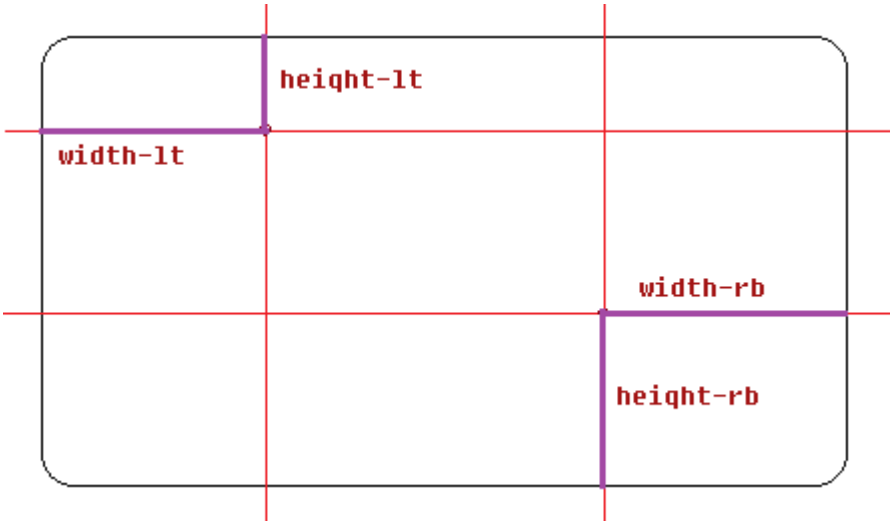
属性名	类型	说明
width	数字	窗口宽度
height	数字	窗口高度
resize	0|1	1 表示窗口可以改变大小
frame	字符串	窗口的 frame 层图片，frame 层是一个可选的半透明 Alpha 图片层，一般设置的这个图片是用于和背

		景图片进行 Alpha 混合，这一层的图片中每个像素都包含了自身颜色和透明度属性，通过透明度属性可以将背景图片进行半透明处理，默认只有主窗口设置了这个 frame 层图片 ,并且默认的 frame 图片是一个透明度渐变的 PNG 图片 ,从顶端的 100%透明到底端的完全不透明
framesize	数字	窗口的 frame 层图片的边框宽度，非九宫格方式有效
width-lt	数字	窗口的 frame 层图片的九宫格左上角位置距离边框的宽度
height-lt	数字	窗口的 frame 层图片的九宫格左上角位置距离边框的高度
width-rb	数字	窗口的 frame 层图片的九宫格右下角位置距离边框的宽度
height-rb	数字	窗口的 frame 层图片的九宫格右下角位置距离边框的高度
bkimg	字符串	窗口的背景图片，如果指定了就使用指定的背景图片，否则使用全局设置的背景图片
crbk	颜色	窗口的背景颜色，如果未指定背景图片，但指定了背景颜色，就使用指定的背景颜色，否则使用全局设置的背景图片
appwin	0|1	此窗口是否会显示在 Windows 任务栏中显示，见下面的截图说明
translucent	数字	窗口的整体透明度，取值范围是 1-255,1 表示全透明，255 表示不透明
crtransparent	颜色	设置窗口的背景透明颜色,RGB 格式
shadow-size	数字	窗口的算法阴影的宽度
img-shadow	图片	窗口的图片阴影使用的图片
shadow-wlt	数字	窗口的阴影层图片的九宫格左上角位置距离边框的宽度
shadow-hlt	数字	窗口的阴影层图片的九宫格左上角位置距离边框的高度

shadow-wrb	数字	窗口的阴影层图片的九宫格右下角位置距离边框的宽度
shadow-hrb	数字	窗口的阴影层图片的九宫格右下角位置距离边框的高度
topmost	0|1	窗口是否显示在所有窗口的最前面（整个桌面范围内）

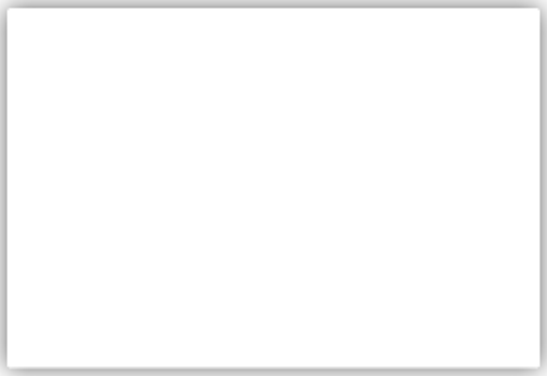
说明：

1）九宫格方式 frame 层的说明：对于复杂的背景 frame 层图片，其所有边框宽度并不是固定的，但一般都可以用九宫格方式来切分，就是把背景 frame 图片横向、纵向各用两条线切分，一共切分成九部分，应用时候四个角的图片大小是按照原始大小应用到窗口中的，其余几部分都会进行拉伸，对于这种方式，只要描述出九宫格的左上角和右下角坐标位置就可以，对应的就是 width-lt、height-lt、width-rb、height-rb 这 4 个属性。



2）appwin 属性的说明：下面截图中右边的任务栏窗口就是因为此窗口设置为 appwin 属性为 1 才会在任务栏中单独显示出来：

3）窗口阴影的说明：窗口支持两种方式的阴影，一种是图片阴影，一种是算法阴影。图片阴影是指定一个九宫格方式的阴影图片，类似下图的图片，需要指定九宫格的坐标，窗口绘制时候会将阴影图片作为一个单独的层画在窗口的下方外围区域。算法阴影是直接通过算法画一个阴影层，需要指定阴影的宽度。如果窗口属性中设置了图片阴影的图片，则优先使用图片阴影方式画阴影层，如果没有指定阴影图片，但设置了阴影宽度，则使用算法阴影方式画阴影层，如果图片和阴影宽度都没有指定，则不会画阴影层。



加了阴影的窗口效果如下：

函数：

函数	是否虚函数	说明
SetXmlFile	否	设置对话框加载的 xml 文件
GetControl	否	根据 ID 或 name 获取对应的控件指针
DoOK	否	对话框的确定
DoCancel	否	对话框的取消
DoClose	否	对话框的关闭
SetControlVisible	是	设置指定控件的可见性
SetControlDisable	是	设置指定控件是否禁用
OpenDlgPopup	否	打开一个弹出框
CloseDlgPopup	否	关闭弹出框

界面示例：

3.7. DUI 弹出窗口

类名：CDlgPopup

控件名：无

说明：弹出窗口类，是菜单、下拉列表等控件的父类，也可以单独使用，用于创建弹出窗口，弹出窗口默认是非激活状态下自动关闭，例如鼠标点击到弹出窗口外面的区域，弹出窗口就会自动关闭，也可以设置为不自动关闭。

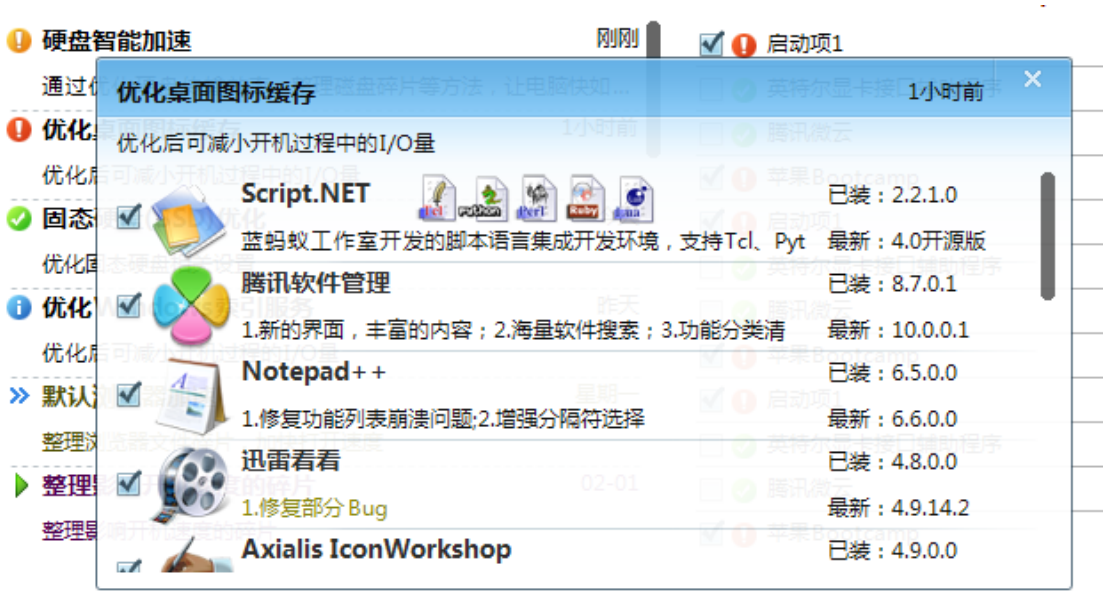
通用的弹出窗口一般通过对话框的 OpenDlgPopup 函数创建。

属性：

属性名	类型	说明
width	数字	窗口宽度
height	数字	窗口高度
bkimg	字符串	窗口的背景图片，如果指定了就使用指定的背景图片，否则使用全局设置的背景图片
bkmod	枚举	窗口背景图片的模式，包括： image – 普通的背景图片 frame – 边框模式 mid – 九宫格模式
framesize	数字	窗口背景是 frame 模式时候，指定边框宽度
width-lt	数字	窗口的背景图片的九宫格左上角位置距离边框的宽度
height-lt	数字	窗口的背景图片的九宫格左上角位置距离边框的高

		度
width-rb	数字	窗口的背景图片的九宫格右下角位置距离边框的宽度
height-rb	数字	窗口的背景图片的九宫格右下角位置距离边框的高度
bkalpha	0|1	背景是否使用 Alpha 通道
translucent	数字	窗口的整体透明度，取值范围是 1-255,1 表示全透明，255 表示不透明
autoclose	0|1	窗口是否自动关闭，默认是鼠标点击到非窗口区域就自动关闭
topmost	0|1	窗口是否显示在所有窗口的最前面（整个桌面范围内）

弹出窗口与界面示例：



3.8. DUI 菜单

类名：CDuiMenu

控件名：无

说明：菜单有两种显示的位置，一种是在窗口顶部某个按钮点击后可以下拉一个菜单，另一种是托盘图标的右键菜单。

窗口中的菜单定义方式是 xml 文件中设置某个按钮的 action 属性，以 menu:开头，后面是菜单的 XML 文件名或 XML 定义名，例如下面这样定义：

```
<imgbtn name="button.menu" pos="-110,0,-77,29" skin="IDB_BT_MENU" tip="菜单" action="menu:mainmenu.xml"/>
```

托盘菜单默认是按照 resource.xml 中定义的 menu_tray 指向的 XML 文件来加载菜单。

两种方式加载的菜单定义 XML 文件格式都是相同的，参考前面 XML 说明章节的示例。

属性：

属性名	类型	说明
width	数字	菜单窗口宽度
item-height	数字	每个菜单项的高度
left	数字	菜单左侧图标区的宽度
sep-height	数字	菜单分隔线的高度
font	字符串	字体
fontwidth	数字	字体宽度
frame-width	数字	菜单项距离边框的宽度
top-height	数字	菜单项顶部距离边框的高度
bottom-height	数字	菜单项底部距离边框的高度
crrowhover	颜色	菜单项背景颜色(鼠标移动到菜单项时候的颜色)，如果不设置则使用默认颜色
img-rowhover	图片	菜单项背景图片(鼠标移动到菜单项时候的背景图片)，优先级比背景颜色高
img-popuparrow	图片	弹出菜单箭头图片

函数：

函数	是否虚函数	说明
LoadXmlFile	否	加载菜单 XML 文件
AddMenu	否	动态添加菜单项
AddSeparator	否	动态添加菜单分隔线
SetItemTitle	否	预设值菜单项的标题
SetItemVisible	否	预设值菜单项的可见性
SetItemDisable	否	预设值菜单项的禁用状态
SetItemCheck	否	预设值菜单项的检查标志

SetMenuPoint	否	刷新所有菜单项的位置信息
GetParentMenu	否	获取父菜单对象
GetHoverMenuItem	否	获取当前激活菜单项对象

界面示例：

3.9. DUI 菜单项

类名：CMenuItem

控件名：menuitem

说明：菜单中加载的每个菜单项的控件，CControlBaseFont 中的所有属性和函数都可以使用。

属性：

属性名	类型	说明
seperator	0|1	是否分隔线
select	0|1	是否选择（如果是 checkbox 或 radiobutton 类型的菜单项，此属性必须设置为 1）
check	0|1	是否处于选中状态

group	字符串	广播按钮所属的组名，相同组名的广播按钮是属于一组的，可以联动，一组中只有一个会处于选中状态
value	字符串	广播按钮的值，一组广播按钮中的多个按钮值是不一样的，当获取这一组广播按钮的值时候，获取的就是选中的按钮的值
menu	字符串	引用其他的菜单的名字（通过资源定义可以找到的菜单的名字），设置了这个属性，则会将对应的菜单嵌入当前菜单中
image	图片	菜单项左侧的小图片，如果是弹出菜单，并且没有设置菜单的 img-popuparrow 属性，则 image 属性表示菜单项右侧的箭头图片
img-count	数字	设置菜单项左侧图片是由几个并列的小图片组成的
taskmsg	0|1	是否通过任务方式执行菜单处理函数，如果弹出菜单的处理函数中有阻塞或等待的操作（例如打开一个对话框），则执行过程中弹出菜单可能会因为失去焦点而将自身的对象删除，这种情况下就需要通过任务方式执行菜单处理函数，任务方式是将操作插入任务队列，由任务队列线程再去执行菜单处理函数

函数：

函数	是否虚函数	说明
SetCheck	否	设置是否选择
GetCheck	否	获取是否选择的状态
IsSeparator	否	判断是否分隔线
SetGroupName	否	设置广播按钮组的名字
GetGroupName	否	获取广播按钮组的名字
GetValue	否	获取广播按钮的值
GetGroupValue	否	获取广播按钮组的值
ResetGroupCheck	否	刷新父控件下面所有同一个组的 RadioButton 控件的状态

3.10. DUI TabCtrl 控件

类名：CDuiTanCtrl

控件名：tabctrl

说明：多个 Tab 页的切换控件，是从 CControlBaseFont 派生的，CControlBaseFont 中的所有属性和函数都可以使用。

属性：

属性名	类型	说明
tab-type	枚举	tabctrl 的模式(此属性必须在其他属性的前面进行设置)： horizontal – 水平方向的 tabctrl，是默认的模式，可以不设置 vertical – 垂直方向的 tabctrl
image	字符串	Tab 页的图标图片，可以是并排多个图片，通过 img-count 可以设置图片个数
img-sep	字符串	Tab 页之间的分割线图片
img-hover	字符串	鼠标移动到 tab 页时候的 tab 状态图片，并排的两张图片，分别是鼠标移动和鼠标点击状态的图片
img-tabbtn	字符串	tab 页签内部按钮图片，并排的四张图片，分别是正常状态、鼠标移动、鼠标点击、禁用状态的图片
tabbtnpos	位置	Tab 页签内部按钮图片显示的位置坐标，必须是 x,y 的形式，表示按钮图片左上角坐标（相对于每一个 tab 页签），例如 10,10。 支持正值和负值，正值表示从 tab 页签左上角开始计算的值，负值表示从 tab 页签右下角开始计算的值，例如-10,10 表示从右边往左 10 像素，从上往下 10 像素的位置。 也可以支持从 tab 页签中间开始计算的坐标，用|表示中间位置，例如|10,|-10 表示横向中间向右 10 像素，纵向中间向上 10 像素的位置。
item-width	数字	每个 tab 页的最大宽度，如果没有设置则按照 img-hover 的图片宽度

		【注：仅水平模式有效】
item-width-min	数字	每个 tab 页的最小宽度，如果设置了这个属性，则 tab 页签的宽度可以自动进行调整，调整的算法是每个 tab 页签宽度实际值是最大宽度和最小宽度之间，根据当前总宽度和页签数量自动进行计算，在插入、删除 tab 页，以及控件位置发生变化时候，会触发 tab 页签宽度的自动更新 【注：仅水平模式有效】
item-height	数字	每个 tab 页的最大高度，如果没有设置则按照 img-hover 的图片高度 【注：仅垂直模式有效】
item-height-min	数字	每个 tab 页的最小高度，如果设置了这个属性，则 tab 页签的高度可以自动进行调整，调整的算法是每个 tab 页签高度实际值是最大高度和最小高度之间，根据当前总高度和页签数量自动进行计算，在插入、删除 tab 页，以及控件位置发生变化时候，会触发 tab 页签高度的自动更新 【注：仅垂直模式有效】
tab-height	数字	Tab 页图标部分高度，如果没有设置则按照 img-hover 的图片高度 【注：仅水平模式有效】
tab-width	数字	Tab 页图标部分宽度，如果没有设置则按照 img-hover 的图片宽度 【注：仅垂直模式有效】
animate	0|1	Tab 页切换时候是否显示动画效果
animate-count	数字	Tab 页切换动画的帧数，默认是 10 帧
crtext	颜色	Tab 页标题的文字颜色
crhover	颜色	鼠标移动时候 Tab 页标题的文字颜色，如果设置为全 0 颜色，则不使用此颜色
crpush	颜色	鼠标点击时候 Tab 页标题的文字颜色，如果设置为全 0 颜色，则不使用此颜色
tab-left-pading	数字	Tab 标签最左侧的空白宽度【注：仅水平模式有效】

tab-right-pading	数字	Tab 标签最右侧的空白宽度【注：仅水平模式有效】
tab-top-pading	数字	Tab 标签最上边的空白宽度【注：仅垂直模式有效】
tab-bottom-pading	数字	Tab 标签最下边的空白宽度【注：仅垂直模式有效】
tab-tip	0|1	是否显示 tab 页签的 tip 信息，默认为显示

函数：

函数	是否虚函数	说明
InsertItem	否	添加一个 tab 页
GetItemIndex	否	根据 tab 页名字获取索引
GetItemInfo	否	根据 tab 页索引获取 tab 页信息
SetItemWidth	否	设置 tab 页签的最大宽度和最小宽度【注：仅水平模式有效】
SetItemHeight	否	设置 tab 页签的最大高度和最小高度【注：仅垂直模式有效】
RefreshItems	否	刷新所有 tab 页，重新计算位置信息
DeleteItem	否	根据索引或名字删除 tab 页
SetSelectItem	否	设置当前激活的 tab 页
SetItemVisible	否	设置 tab 页是否可见
LoadTabXml	否	从 XML 文件加载 tab 页

界面示例(水平模式和垂直模式)：

Tab 页签按钮的说明：

每个 Tab 页签中可以放一个图片按钮，例如实现用于关闭 Tab 页签的按钮，通过 img-tabbtn

属性可以设置这个图片按钮的图片，每个 tab 页签都是使用相同的图片，通过 tabbtnpos 属性可以设置按钮图片相对于每个 Tab 页签的位置，如果不设置位置，默认会显示在 Tab 页签的右上角，使用页签按钮实现的带关闭按钮的 Tab 页效果如下：

页签按钮点击之后会发送一个消息到事件处理类，消息类型是 MSG_CONTROL_BUTTON，消息中的 wParam 参数表示 tab 页签的索引。

Tab 页签文字和图片的对齐方式：

如果只有文字或图片，则使用设置的对齐方式，默认是居中对齐。如果有文字和图片，则水平方向使用设置的对齐方式，垂直方向图片是从 0 开始画图，文字在图片的下方。

3.11. DUI Tab 控件

类名：CDuiTanCtrl

控件名：tab

说明：Tab 页控件的下层控件，每个 tab 表示一个页面。

属性：

属性名	类型	说明
name	字符串	可以根据名字获取到 tab 页对应的 DuiPanle 控件对象
title	字符串	Tab 页标题
tip	字符串	Tab 页的 tip 信息，如果设置了 tab 页标题，并且标题显示不全，则使用 tip 页标题作为 tip，否则使用 tip 属性作为 tip 信息
active	true|false	此 Tab 页是否激活
show	0|1 或 true|false	此 Tab 页是否显示
visible	0|1 或 true|false	和 show 属性相同
image	字符串	此 tab 页的页签图片，是一个 3 张横向切片图组成图片，3 张图分别表示正常、鼠标移动、鼠标点击 3 种状态下的 tab 页签
img-index	数字	如果是多张图片组成的大图，则表示图片的索引

img-count	数字	如果是多张图片组成的大图，则表示图片的个数
div	字符串	如果一个 tab 页的内容想写在一个单独的 xml 文件中，可以在此处定义 xml 文件名
pos	字符串	Tab 页的位置信息字符串
outlink	0|1 或 true|false	表示此 tab 页是否是外部链接，外部链接是指点击之后打开一个网页或独立的窗口或程序，对于外部链接，是不会显示鼠标点击到此 tab 页的状态的
action	字符串	点击此 tab 页时候执行的动作
scroll	0|1	此 tab 页是否支持滚动

3.12. DUI Panel 控件

类名：CDuiPanel

控件名：div

说明：Panel 控件是一种虚拟的容器控件，本身不会有任何显示界面，但可以在下层包含若干其他的控件，通过设置 Panel 的可见性可以控制下层所有控件的可见性。Panel 类是从 CControlBaseFont 派生的，所以基类控件具有的属性和函数也可以使用。

属性：

属性名	类型	说明
xml	字符串	Panel 控件可以通过加载一个 xml 文件来加载下层控件，此处是指 xml 文件名
img-scroll	字符串	垂直滚动条图片
img-scrollh	字符串	水平滚动条图片
scroll-width	数字	滚动条宽度
scroll	0|1	是否允许滚动
plugin	0|1	DuiVision 界面插件的 release 版本动态库文件，具体插件说明请参考插件章节，主程序是 release 版本编译时候此属性才有效
Plugin-debug	0|1	DuiVision 界面插件的 debug 版本动态库文件，具体插件说明请参考插件章节，主程序是 debug 版本编译时候此属性才有效

函数：

函数	是否虚函数	说明
LoadXmlFile	否	加载 XML 文件
SetVirtualHeight	否	设置 div 的虚拟高度
SetVirtualWidth	否	设置 div 的虚拟宽度
SetEnableScroll	否	设置 div 是否允许滚动
GetEnableScroll	否	获取 div 是否允许滚动

说明：

1、Panel 的虚拟高度和宽度计算方法是遍历容器中的所有子控件，找到底部和右边的最大值作为虚拟高度和宽度，也可以通过 API 人工设置虚拟高度和宽度。

3.13. DUI 布局控件

类名：CDuiLayout

控件名：layout

说明：layout 控件是一种用于横向和纵向布局和分隔窗口的控件，layout 控件可以将一个 Panel 按照横向或纵向分割为两个或多个子 Panel，分割时候可以指定子 Panel 的宽度或高度，并且可以显示分隔线和分隔滑块。Panel 类是从 CDuiPanel 派生的，所以基类控件具有的属性和函数也可以使用。

属性：

属性名	类型	说明
layout-type	枚举	horizontal 表示对父 Panel 纵向分割为多个窗口，分隔线为水平方向；vertical 表示对父 Panel 横向分割为多个窗口，分隔线为垂直方向
img-split	字符串	分隔线图片
img-thumb	字符串	滑块图片
split-width	数字	分隔线宽度
split-imm	0|1	鼠标拖动分割线或滑块时候是否即时改变子 Panel 的位置

Layout 控件的下层子 Panel xml 节点名为 layout-div，layout-div 的属性如下表：

属性名	类型	说明
div-pos	数字	子 Panel 的初始宽度或高度，支持正值、负值和百分数，正值表示从父控件左上角开始计算的值，负值表示从父控件右下角开始计算的值，百分数表示

		按照父控件宽度或高度的百分比。 如果人工拖动了分割线的宽度或高度，则后续的宽度或高度以拖动后的为准。
min-pos	数字	子 Panel 的宽度或高度的最小值，拖动时候不能比这个值更小，不设置表示无限制
max-pos	数字	子 Panel 的宽度或高度的最大值，拖动时候不能比这个值更大，不设置表示无限制

Xml 示例：

```
<layout name="split_1" layout-type="vertical" pos="0,0,-0,-0"
      img-split="skin:IDB_SPLIT_VERT" split-width="2" split-imm="1"
      img-thumb="skin:IDB_SPLIT_THUMB_VERT" >
      <layout-div name="layout-1" div-pos="300" min-pos="200" max-pos="500" >
            ...
      </layout-div>
      <layout-div name="layout-2">
            ...
      </layout-div>
</layout>
```

函数：

函数	是否虚函数	说明
GetLayoutItemInfo	否	获取指定子 Panel 的信息
GetLayoutPos	否	计算某个子 Panel 实际的宽度或高度
SetSplitPos	否	设置某个子 Panel 实际的宽度或高度

界面示例：

3.14. DUI 文字控件

类名：CDuiText

控件名：text

说明：显示指定的文字，支持文字中指定的字符串设置不同的颜色，是从 CControlBaseFont 派生的，所以基类控件具有的属性和函数也可以使用。

属性：

属性名	类型	说明
crtext	颜色	文字颜色
crhover	颜色	鼠标移动时候的颜色
crshadow	颜色	文字阴影颜色
crback	颜色	背景颜色
crmark	颜色	特殊显示的文字颜色
mask	字符串	特殊显示的字符串
img-scroll	字符串	滚动条图片
scroll-width	数字	滚动条宽度
bk-transparent	数字	背景透明度，未指定背景颜色时才有效，0 表示全透明，100 表示不透明，默认为 0

函数：

函数	是否虚函数	说明
SetTitleMarkText	否	设置整个字符串和特殊显示部分字符串

界面示例：

3.15. DUI 图片控件

类名：CDuiPicture

控件名：img

说明：显示指定的图片，是从 CControlBaseFont 派生的，所以基类控件具有的属性和函数也可以使用。

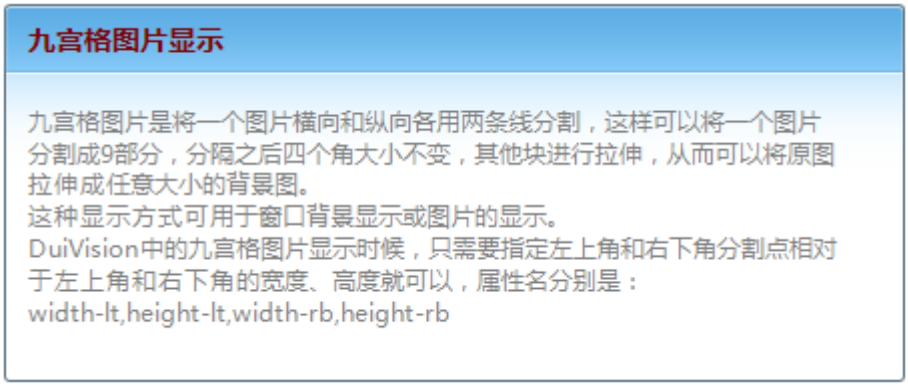
属性：

属性名	类型	说明
mode	枚举	图片的显示模式，集中模式分别是： normal - 正常模式显示图片 tile - 平铺模式显示 extrude - 拉伸模式显示 frame - 显示图片边框 mid - 九宫格方式显示图片
framesize	数字	显示图片边框的模式下，表示边框的宽度
width-lt	数字	九宫格模式下的九宫格左上角位置距离边框的宽度
height-lt	数字	九宫格模式下的九宫格左上角位置距离边框的高度
width-rb	数字	九宫格模式下的九宫格右下角位置距离边框的宽度
height-rb	数字	九宫格模式下的九宫格右下角位置距离边框的高度

函数：

函数	是否虚函数	说明
SetShowMode	否	设置图片显示模式
SetShowModeMID	否	设置九宫格方式的图片显示模式

界面示例：



3.16. DUI 动画图片控件

类名：CDuiAnimateImage

控件名：animateimg

说明：显示动画图片，是从 CControlBaseFont 派生的，所以基类控件具有的属性和函数也可以使用。此控件支持两种动画图片：一种是图片按照横向若干张小图片拼接的方式，小图片的宽度必须相同，控件可以设置小图片的个数，控件内可以自动启动一个定时器，按照顺序循环定时显示小图片，达到动画的效果；另一种是设置 GIF 动画图片，按照 GIF 动画图片中的帧数和延迟时间进行动画图片的显示。

属性：

属性名	类型	说明
image	图片	参见文字控件基础类中的 image 定义，如果不是 GIF 图片，就需要指定这个属性
img-gif	图片	如果是 GIF 图片，需要指定这个属性，如果指定了这个属性 ,再指定 image 属性也没有意义 ,img-gif 属性的用法和 image 类似
index	数字	当前显示第几张图片
maxindex	数字	小图片的个数，如果是 GIF 图片，可以不指定，会自动按照 GIF 的帧数
timer-count	数字	图片切换的定时器周期数，每个周期的时间是 30 毫秒，也就是图片切换的时间是多少个 30 毫秒，如果是 GIF 图片，此属性没有意义，会按照 GIF 中每一帧的动画时间进行定时，GIF 的定时时间也会按照 30 毫秒向上取整
run	0|1	表示是否启动动画，启动之后动画图片可以自己按

		照设置的定时周期进行变化

函数：

函数	是否虚函数	说明
SetRun	否	设置是否启动动画
SetTimerCount	否	设置动画的图片切换周期

界面示例（动画图片的原图）：

上面这个图片是一个包含 4 个小图片的动画图片。

3.17. DUI 按钮控件

类名：CDuiButton、CImageButton

控件名：button、imgbtn

说明：显示按钮，是从 CControlBaseFont 派生的，所以基类控件具有的属性和函数也可以使用。按钮的背景图片是由 4 张切图拼接成的大图片，分别是普通状态、鼠标移动状态、鼠标按下状态、禁用状态的图片，背景图片设置可以通过 skin 或 image 属性设置。

按钮控件有两种控件名，button 和 imgbtn，唯一的差别是 imgbtn 默认会启用渐变效果定时器，并且渐变效果的帧数默认设置为 10 帧。

属性：

属性名	类型	说明
crtext	颜色	按钮文字的颜色
animate	0|1	是否启用渐变效果定时器
maxindex	0|1	渐变效果的最大帧数
img-btn	字符串	图片按钮的图片文件
showfocus	0|1	控件处于焦点状态时是否显示焦点虚线框（默认是显示）

函数：

函数	是否虚函数	说明

SetMaxIndex	否	设置渐变效果的最大帧数

界面示例：

3.18. DUI 文字按钮控件

类名：CTextButton

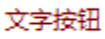
控件名：textbtn

说明：文字按钮是从 CControlBaseFont 派生的，所以基类控件具有的属性和函数也可以使用。此控件显示效果是一段可点击的文字，文字可以设置普通状态、鼠标移动状态、鼠标按下状态、禁用状态的颜色。

属性：

属性名	类型	说明
crtext	颜色	按钮文字的颜色
crhover	颜色	鼠标移动时候的颜色
crpush	颜色	鼠标点击时候的颜色
crdisable	颜色	禁用状态的颜色

界面示例：



3.19. DUI 链接按钮控件

类名：CLinkButton

控件名：linkbtn

说明：链接按钮是从 CControlBaseFont 派生的，所以基类控件具有的属性和函数也可以使用。此控件显示效果是一段可点击的文字，文字可以设置普通状态、鼠标移动状态、鼠标按下状态、禁用状态的颜色，文字点击之后可以打开一个网页或文件。

属性：

属性名	类型	说明
crtext	颜色	按钮文字的颜色
crhover	颜色	鼠标移动时候的颜色
crpush	颜色	鼠标点击时候的颜色
crdisable	颜色	禁用状态的颜色
href	字符串	链接 URL

函数：

函数	是否虚函数	说明
SetLink	否	设置链接 URL
GetLink	否	获取链接 URL

界面示例：

点击打开链接

3.20. DUI 隐藏按钮控件

类名：CHideButton

控件名：hidebtn

说明：文字按钮是从 CControlBaseFont 派生的，所以基类控件具有的属性和函数也可以使用。此控件显示效果是一段文字右边有一段默认隐藏的文字链接，正常情况下看不到右边的文字链接，鼠标移动到左边的文字时候才能看到右边的文字链接。文字可以设置普通状态、鼠标移动状态、鼠标按下状态、禁用状态的颜色。

属性：

属性名	类型	说明
crtext	颜色	按钮文字的颜色
crhover	颜色	鼠标移动时候的颜色
crpush	颜色	鼠标点击时候的颜色
crdisable	颜色	禁用状态的颜色
crtip	颜色	隐藏链接的文字颜色
text	字符串	隐藏链接的字符串显示内容

界面示例：

隐藏按钮 点击

3.21. DUI 检查框控件

类名：CCheckButton

控件名：chkbtn

说明：显示检查框，是从 CControlBaseFont 派生的，所以基类控件具有的属性和函数也可以使用。

属性：

属性名	类型	说明
check	true|false	检查框是否处于选择状态
crtext	颜色	文字颜色
showfocus	0|1	控件处于焦点状态时是否显示焦点虚线框（默认是显示）

函数：

函数	是否虚函数	说明
SetCheck	否	设置检查框的值
GetCheck	否	获取检查框的值
SetTextColor	否	设置文字颜色

DUI 消息：

消息 ID	函数	说明
MSG_BUTTON_UP	OnControlLButtonUp	鼠标点击检查框，放开时候会发送此消息
MSG_BUTTON_CHECK	OnControlKeyDown	键盘触发快捷键时候，会发送此消息

界面示例：

3.22. DUI 广播按钮控件

类名：CDuiRadioButton

控件名：radiobtn

说明：显示广播按钮，是从 CControlBaseFont 派生的，所以基类控件具有的属性和函数也可以使用。

属性：

属性名	类型	说明
check	true|false	广播按钮是否处于选中状态
crtext	颜色	文字颜色
group	字符串	广播按钮所属的组名，相同组名的广播按钮是属于一组的，可以联动，一组中只有一个会处于选中状态
value	字符串	广播按钮的值，一组广播按钮中的多个按钮值是不一样的，当获取这一组广播按钮的值时候，获取的就是选中的按钮的值
showfocus	0|1	控件处于焦点状态时是否显示焦点虚线框（默认是显示）

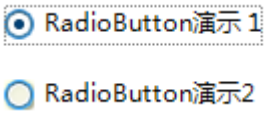
函数：

函数	是否虚函数	说明
SetCheck	否	设置检查框的值
GetCheck	否	获取检查框的值
SetGroupName	否	设置广播按钮组的名字
GetGroupName	否	获取广播按钮组的名字
GetValue	否	获取广播按钮的值
GetGroupValue	否	获取广播按钮组的值
ResetGroupCheck	否	刷新父控件下面所有同一个组的 RadioButton 控件的状态
SetTextColor	否	设置文字颜色

DUI 消息：

消息 ID	函数	说明
MSG_BUTTON_UP	OnControlLButtonUp	鼠标点击广播按钮，放开时候会发送此消息
MSG_BUTTON_CHECK	OnControlKeyDown	键盘触发快捷键时候，会发送此消息

界面示例：



3.23. DUI 进度条控件

类名：CDuiProgress

控件名：progress

说明：显示进度条，是从 CControlBaseFont 派生的，所以基类控件具有的属性和函数也可以使用。

属性：

属性名	类型	说明
value	数字	进度值，最大值不能超过 max-value 的值
max-value	数字	最大的进度值，默认为 100
timer-count	数字	定时器定时多少次，自动增长的进度值加 1
run	true|false	进度条是否自动运行（循环增长）
img-back	字符串	背景图片
img-fore	字符串	前景图片
head-len	数字	左右两侧圆角部分长度，如果此长度不为 0，表示头部是圆角的进度条，画图时候头部圆角按照实际图片画图，其余部分则拉伸显示
show-text	0|1	是否显示进度条的文字，如果显示文字，可以通过 align 和 valign 属性设置文字的对齐方式，显示的文字内容是前缀+当前进度值，如果进度最大值是 100，则自动在当前进度值后面加%
crtext	颜色	进度条文字的颜色
title	字符串	进度条文字的前缀，例如 title 设置为”进度：”，则显示出来的整体的进度条文字是”进度：xx%”

函数：

函数	是否虚函数	说明
SetProgress	否	设置进度条的值

GetProgress	否	获取进度条的值
SetMaxProgress	否	设置进度条的最大值
GetMaxProgress	否	获取进度条的最大值
SetRun	否	设置自动运行

界面示例：

3.24. DUI 滑动条控件

类名：CDuiSlider

控件名：slider

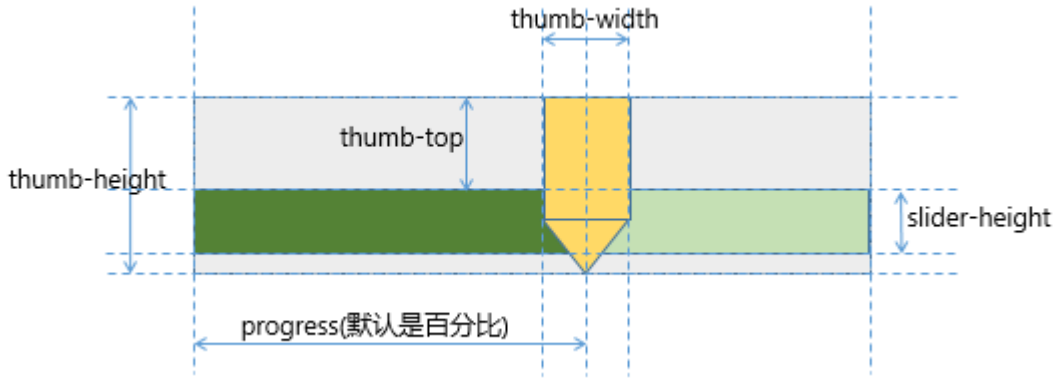
说明：显示滑动条，是从 CControlBaseFont 派生的，所以基类控件具有的属性和函数也可以使用。

属性：

属性名	类型	说明
value	数字	进度值，最大值不能超过 max-value 的值
max-value	数字	最大的进度值，默认为 100
img-back	字符串	滑动条背景图片
img-fore	字符串	滑动条前景图片，显示当前进度的区域
img-thumb	字符串	滑动块图片，由 4 个横向小图片组成，鼠标移动时候可以改变状态，鼠标可以拖动滑动块，拖动时候会发送 MSG_SCROLL_CHANGE 消息
slider-height	字符串	滑动条高度
thumb-width	字符串	滑动块宽度，如果不设置则按照滑动块图片的宽度，设置之后可以按设置值拉伸
thumb-height	字符串	滑动块高度，如果不设置则按照滑动块图片的高度，设置之后可以按设置值拉伸
thumb-top	字符串	滑动块顶部和滑动条顶部的距离
head-len	数字	左右两侧圆角部分长度，如果此长度不为 0，表示头部是圆角的进度条，画图时候头部圆角按照实际

		图片画图，其余部分则拉伸显示
show-text	0|1	是否显示进度条的文字，如果显示文字，可以通过 align 和 valign 属性设置文字的对齐方式，显示的文字内容是前缀+当前进度值，如果进度最大值是 100，则自动在当前进度值后面加%
crtext	颜色	进度条文字的颜色
title	字符串	进度条文字的前缀，例如 title 设置为”进度：” ，则显示出来的整体的进度条文字是”进度：xx%”

几个关键位置属性的含义：



函数：

函数	是否虚函数	说明
SetProgress	否	设置进度条的值
GetProgress	否	获取进度条的值
SetMaxProgress	否	设置进度条的最大值
GetMaxProgress	否	获取进度条的最大值

界面示例：

3.25. DUI 输入框控件

类名：CDuiEdit

控件名：edit

说明：输入框控件，是从 CControlBaseFont 派生的，所以基类控件具有的属性和函数也可以使用。

属性：

属性名	类型	说明
password	0|1	是否是密码输入框
multiline	0|1	是否多行输入框
wantreturn	0|1	是否允许回车换行，如果设置为 0，则会屏蔽掉此输入控件的回车换行操作
autohscroll	0|1	水平宽度超出是否可以滚动显示
autovscroll	0|1	垂直高度超出是否可以滚动显示
hscroll	0|1	是否显示水平滚动条
vscroll	0|1	是否显示垂直滚动条
number	0|1	是否只能输入数字
readonly	0|1	是否只读
maxchar	数字	最多可以输入多少个字符
left-image	字符串	输入框左边显示的小图片
small-image	字符串	输入框右边显示的小图片
crback	颜色	输入框的背景颜色

函数：

函数	是否虚函数	说明
GetEditText	否	获取编辑框文字
SetSmallBitmap	否	设置小图片
SetBackColor	否	设置输入框的背景颜色

界面示例：

3.26. DUI rich 输入框控件

类名：CDuiRichEdit

控件名：richedit

说明：rich 输入框控件，是从 CDuiPanel 派生的，所以基类控件具有的属性和函数也可以使用。

属性：

属性名	类型	说明
password	0|1	是否是密码输入框
multiline	0|1	是否多行输入框
wantreturn	0|1	是否允许输入回车键进行换行
wantctrlreturn	0|1	是否允许输入 Ctrl+回车键进行换行
hscrollbar	0|1	是否显示水平滚动条
vscrollbar	0|1	是否显示垂直滚动条
autohscroll	0|1	水平宽度超出是否可以滚动显示
autovscroll	0|1	垂直高度超出是否可以滚动显示
number	0|1	是否只能输入数字
readonly	0|1	是否只读
maxchar	数字	最多可以输入多少个字符
rich	0|1	是否支持 rtf 格式的字符串
wordwrap	0|1	是否支持单词自动换行
file	0|1	加载的 rtf 文件名
startchar	数字	选择块的开始位置
endchar	数字	选择块的结束位置

crtext	颜色	文字颜色
small-image	字符串	输入框右边显示的小图片
left-image	字符串	输入框左侧显示的小图片

函数：

函数	是否虚函数	说明
GetText	否	获当前文字

界面示例：

3.27. DUI 下拉列表控件

类名：CDuiComboBox

控件名：combobox

说明：下拉列表控件，是从 CDuiEdit 派生的，所以基类控件具有的属性和函数也可以使用。

属性：

属性名	类型	说明
value	字符串	下拉列表当前选择项的值
head-image	字符串	下拉列表项中左边部分图片的掩码图片
del-image	字符串	下拉列表项中每一行的删除按钮，如果没有设置则不能由用户删除列表项
xml	字符串	指定下拉列表内容定义的 xml 文件，xml 文件中的下拉列表项描述的格式和非独立 xml 文件方式定义的格式完全相同，如果不指定 xml 文件，则使用当前控件属性的下级节点定义的 xml 内容

crtext	颜色	列表项的颜色，如果是名字和描述两行内容，则表示名字的颜色
crdesc	颜色	名字和描述两行内容的情况下，表示描述部分的颜色
crhover	颜色	当前选中的行（鼠标移动的行）的颜色

Combobox 的下来列表内容有三种定义的方式，分别是在控件 xml 的下级节点定义、通过独立 xml 文件定义、调用控件的 AddItem 函数通过程序添加，xml 定义方式中下拉项的 xml 节点名是 item，节点属性如下表：

属性名	类型	说明
name	字符串	列表项名字
desc	字符串	列表项描述。 如果所有列表项的描述字符串都为空，则下拉列表自动调整为单行显示； 否则当鼠标移动到某一项时候此列表项显示为两行（名字和描述各一行），其他列表项则只显示一行（名字）
value	字符串	列表项的值
image	字符串	列表项中左边部分图片
crtext	颜色	列表项的颜色，如果是名字和描述两行内容，则表示名字的颜色
crdesc	颜色	名字和描述两行内容的情况下，表示描述部分的颜色

函数：

函数	是否虚函数	说明
SetComboValue	否	设置下拉列表的值
GetComboValue	否	获取下拉列表的值
GetItemCount	否	获取下拉列表项个数
AddItem	否	添加列表项
ClearItems	否	清空列表项

界面示例：

3.28. DUI 列表控件

类名：CDuiListCtrl

控件名：listctrl

说明：列表控件，是从 CDuiPanel 派生的，所以基类控件具有的属性和函数也可以使用。

属性：

属性名	类型	说明
img-scroll	字符串	滚动条图片
img-sep	字符串	行分隔线图片
img-check	字符串	行左侧的检查框图片
font-title	字符串	标题行字体
crtext	颜色	正文文字颜色
crhover	颜色	鼠标移动时候的文字颜色，如果设置为全 0 颜色，则不使用此颜色
crpush	颜色	鼠标点击的行的文字颜色，如果设置为全 0 颜色，则不使用此颜色
crtitle	颜色	标题颜色
crsep	颜色	分割线颜色，如果设置为全 0 颜色，则不显示分隔线
crrowhover	颜色	鼠标移动时候行的背景颜色，可以设置透明度，如

		果设置为全 0 颜色，则不显示鼠标移动时候的行背景颜色
crrowcurrent	颜色	当前行的背景颜色，可以设置透明度，如果设置为全 0 颜色，则不显示当前行的行背景颜色
img-width	数字	图片宽度
row-height	数字	行高度
bk-transparent	数字	背景透明度，0-100，0 表示全透明，100 表示不透明
wrap	0|1	文字是否可以换行
down-row	0|1	是否允许当前行特殊显示，当前行是指当前鼠标点击选择的行或者通过键盘上下箭头移动选择的行，可以通过 crpush 和 crrowcurrent 属性设置当前行的文字颜色和背景颜色
row-tip	0|1	是否显示行的 tooltip(如果 title 部分超出则 tip 显示 title，否则如果 content 部分超出则 tip 显示 content 内容)

行属性：

属性名	类型	说明
id	字符串	行 ID，可用于定位
title	字符串	行标题文字
content	字符串	行正文文字
time	字符串	行的时间显示内容
check	0|1	是否显示行左侧的检查框
image	字符串	行左侧的图片
right-img	字符串	行右侧的图片
crtext	颜色	正文文字颜色（如果颜色为全 0，则不使用此颜色）
crback	颜色	行背景颜色（如果颜色为全 0，则不使用此颜色）
link1	字符串	链接文字 1
linkaction1	字符串	链接 1 的执行动作，参考控件基础类中 action 的用法
link2	字符串	链接文字 2

linkaction2	字符串	链接 2 的执行动作 ,参考控件基础类中 action 的用法

函数：

函数	是否虚函数	说明
InsertItem	否	添加行
GetItemInfo	否	根据行索引号获取行的数据结构
ClearItems	否	清空所有行
EnsureVisible	否	将指定的行滚动到可见范围

界面示例：

3.29. DUI 表格控件

类名：CDuiGridCtrl

控件名：gridctrl

说明：表格控件，是从 CDuiPanel 派生的，所以基类控件具有的属性和函数也可以使用。

属性：

属性名	类型	说明
img-scroll	字符串	垂直滚动条图片
img-scrollh	字符串	水平滚动条图片
img-header	字符串	标题行背景图片
img-header-sort	字符串	标题行的列排序状态图片（表示升序还是降序），由 8 个小图片组成，前 4 个是降序图片，后 4 个是升序图片，目前只用到了两组小图片中的第一个图片
img-colsep	字符串	列分隔线图片
img-sep	字符串	行分隔线图片
img-check	字符串	行左侧的检查框图片
font-title	字符串	标题行字体
valign-header	枚举	标题行文字的垂直对齐模式，top、middle、bottom
align-header	枚举	标题行文字的水平对齐模式，left、center、right
crheader	颜色	标题行文字颜色
crtext	颜色	正文文字颜色
crhover	颜色	鼠标移动时候的文字颜色，如果设置为全 0 颜色，则不使用此颜色
crpush	颜色	鼠标点击的行的文字颜色，如果设置为全 0 颜色，则不使用此颜色
crtitle	颜色	标题颜色
crsep	颜色	分割线颜色，如果设置为全 0 颜色，则不显示分隔线
crrowhover	颜色	鼠标移动时候行的背景颜色，可以设置透明度，如果设置为全 0 颜色，则不显示鼠标移动时候的行背景颜色
crrowcurrent	颜色	当前行的背景颜色，可以设置透明度，如果设置为全 0 颜色，则不显示当前行的行背景颜色
img-width	数字	图片宽度
row-height	数字	行高度

bk-transparent	数字	背景透明度，0-100，0 表示全透明，100 表示不透明
row	子节点	表示表格的行节点，具体属性参见行节点表
left-pos	数字	左侧起始位置
wrap	0|1	文字是否可以换行
down-row	0|1	是否允许当前行特殊显示，当前行是指当前鼠标点击选择的行或者通过键盘上下箭头移动选择的行，可以通过 crpush 和 crrowcurrent 属性设置当前行的文字颜色和背景颜色
grid-tip	0|1	是否显示单元格的 tip 信息，默认为显示
column-sep	0|1	是否显示内容部分的列分隔线
modify-column-width	0|1	是否允许鼠标拖动修改列宽度
sort-click	0|1	是否允许点击标题行某一列对此列进行排序

列属性：

属性名	类型	说明
title	字符串	列标题（暂不支持）
crtext	颜色	列文字颜色（暂不支持）
width	数字	列宽度
valign	枚举	文字的垂直对齐模式，top、middle、bottom
align	枚举	文字的水平对齐模式，left、center、right

行属性：

属性名	类型	说明
id	字符串	行 ID，可用于定位
check	0|1	是否显示行左侧的检查框
image	字符串	行左侧的图片
right-img	字符串	行右侧的图片
crtext	颜色	正文文字颜色，没有设置单元格颜色则使用此颜色定义（如果颜色为全 0，则不使用此颜色）
crback	颜色	行背景颜色（如果颜色为全 0，则不使用此颜色）
item	子节点	表示此行的单元格节点，具体属性参见单元格节点

		表

单元格属性：

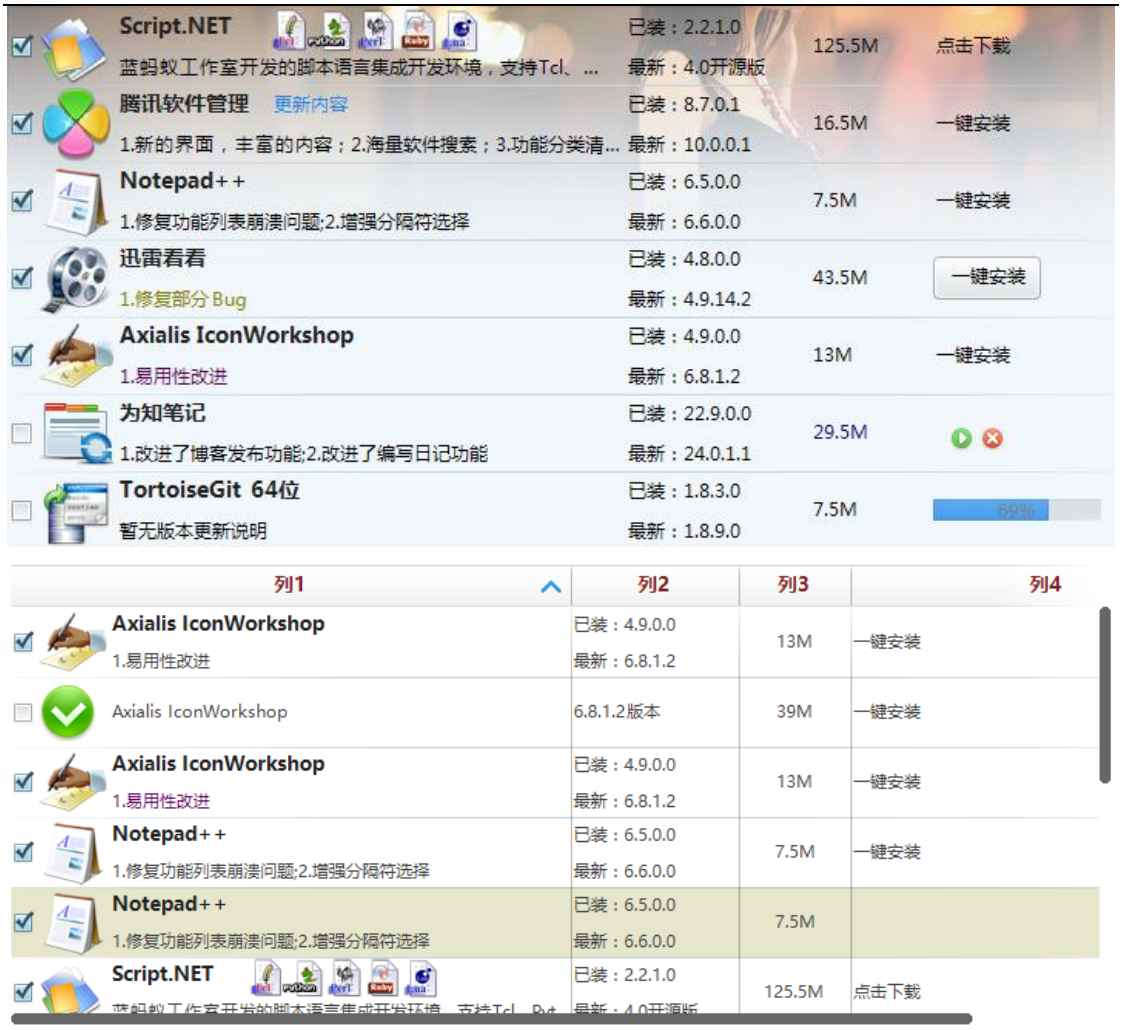
属性名	类型	说明
title	字符串	单元格标题
content	字符串	单元格内容
image	字符串	单元格图片
link	字符串	单元格链接文字
linkaction	字符串	单元格链接动作
crtext	颜色	正文文字颜色（如果颜色为全 0，则不使用此颜色）
font-title	0|1	是否使用标题字体显示单元格文字

单元格可以添加子控件，XML 中只要在单元格的 item 节点下面添加控件子节点就可以。

函数：

函数	是否虚函数	说明
InsertRow	否	添加行
GetItemInfo	否	根据行索引号获取行的数据结构
ClearItems	否	清空所有行
AddSubItemControl	否	添加单元格子控件
DeleteSubItemControl	否	删除单元格子控件
EnsureVisible	否	将指定的行滚动到可见范围
SetSortCompareFunction	否	设置表格的自定义排序比较函数指针

界面示例：



说明：

1、表格排序支持使用自定义排序比较算法函数，方法是调用 SetSortCompareFunction 函数设置一个自定义比较算法函数，比较算法函数的定义如下：

typedef int (CALLBACK *PFN_GRIDCTRL_COMPARE)(GridItemInfo* pItem1, GridItemInfo* pItem2);

此函数会传入两个参数，分别是用于比较的两个单元格信息对象，自定义函数中可以通过这两个参数获取到单元格的各种信息，例如想针对单元格的内容字符串做排序，就从单元格信息中获取到内容字符串来比较，此函数的返回结果表示两个单元格的比较结果，如果返回值小于 0 表示第一个单元格小于第二个单元格 ,如果大于 0 表示第一个单元格大于第二个单元格，如果等于 0 表示两个单元格相等。

如果不设置此自定义排序比较函数，则按照默认的排序规则进行排序，也就是按照单元格的 title 字符串进行比较排序。

3.30. DUI 树控件

类名：CDuiTreeCtrl

控件名：treectrl

说明：树控件，是从 CDuiPanel 派生的，所以基类控件具有的属性和函数也可以使用。

属性：

属性名	类型	说明
img-scroll	字符串	滚动条图片
img-sep	字符串	行分隔线图片
img-check	字符串	节点左侧的选择框图片
img-collapse	字符串	行缩放图片
img-toggle	字符串	树节点收缩图片
font-title	字符串	标题行字体
crtext	颜色	正文文字颜色
crhover	颜色	鼠标移动时候的文字颜色
crpush	颜色	鼠标点击的行的文字颜色
crtitle	颜色	标题颜色
crsep	颜色	分割线颜色
crrowhover	颜色	鼠标移动时候行的背景颜色，可以设置透明度，如果设置为全 0 颜色，则不显示鼠标移动时候的行背景颜色
crrowcurrent	颜色	当前行的背景颜色，可以设置透明度，如果设置为全 0 颜色，则不显示当前行的行背景颜色
img-width	数字	图片宽度
row-height	数字	行高度
bk-transparent	数字	背景透明度，0-100，0 表示全透明，100 表示不透明
node	子节点	树节点
left-pos	数字	左侧起始位置
wrap	0|1	文字是否可以换行
down-row	0|1	是否允许当前行特殊显示，当前行是指当前鼠标点击选择的行或者通过键盘上下箭头移动选择的行，

		可以通过 crpush 和 crrowcurrent 属性设置当前行的文字颜色和背景颜色
grid-tip	0|1	是否显示单元格的 tip 信息

树节点属性：

属性名	类型	说明
id	字符串	行 ID，可用于定位
check	0|1	是否显示行左侧的检查框
image	字符串	行左侧的图片
right-img	字符串	行右侧的图片
crtext	颜色	正文文字颜色，没有设置单元格颜色则使用此颜色定义（如果颜色为全 0，则不使用此颜色）
crback	颜色	节点行背景颜色（如果颜色为全 0，则不使用此颜色）
collapse	0|1	当前是否处于收缩状态
item	子节点	表示此行的单元格节点，具体属性参见单元格节点表

单元格属性：

属性名	类型	说明
title	字符串	单元格标题
content	字符串	单元格内容
image	字符串	单元格图片
link	字符串	单元格链接文字
linkaction	字符串	单元格链接动作
crtext	颜色	正文文字颜色（如果颜色为全 0，则不使用此颜色）
font-title	0|1	是否使用标题字体显示单元格文字
collapse	0|1	此单元格是否可以用于收缩和展开子节点

单元格可以添加子控件，XML 中只要在单元格的 item 节点下面添加控件子节点就可以。

函数：

函数	是否虚函数	说明

InsertNode	否	添加树节点
GetNodeInfo	否	根据节点句柄获取节点的数据结构
ClearNodes	否	清空所有节点
AddSubItemControl	否	添加单元格子控件
DeleteSubItemControl	否	删除单元格子控件
EnsureVisible	否	将指定的节点滚动到可见范围，如果节点处于收缩状态可以自动展开

界面示例：

3.31. DUI 原生 Windows 控件

类名：CDuiNativeWnd

控件名：nativewnd

说明：Windows 原生控件，用于 DuiVision 库中没有的控件，但标准 windows 控件有，或者有自己扩展的 Windows 控件（从 CWnd 类派生的或者其他的 Windows 控件都可以），则可以通过原生控件的方式来使用，使用方法是在 xml 中定义原生控件的名字、位置信息，然后再代码中创建原生控件的对象，创建之后把控件对象或者窗口句柄传递给 CDuiNativeWnd 对象就可以。

属性：

属性名	类型	说明
delaycreate	0|1	是否延迟创建控件

函数：

函数	是否虚函数	说明

GetNativeHWnd	否	获取原生控件的 Windows 窗口句柄
GetNativeWnd	否	获取原生控件的 CWnd 对象指针
GetPaintHWnd	否	获取父对话框的 Windows 窗口句柄
GetPaintWnd	否	获取父对话框的 CWnd 对象指针
SetNativeHWnd	否	设置原生控件的 Windows 窗口句柄
SetNativeWnd	否	设置原生控件的 CWnd 对象指针(设置之后在控件析构时候可以自动删除此 CWnd 对象)

示例：

参考 Demo 程序中输入框控件演示页面中的原生输入框控件演示。

注意事项：

1、原生控件在动态创建之后可能不会显示出来，因为默认是没有调用原生控件的 ShowWindow 进行显示的，有两种方法可以显示出来，一种方法是调用控件的 SetVisible 方法，另一种方法是调用 Windows 的 ShowWindow 函数，第一个参数传入原生控件的句柄。如果原生控件位于 tabctrl 的 tab 页当中，则切换页面时候会自动将原生控件显示或隐藏，如果 tabctrl 只有一个页面，或者原生控件位于第一个页面，则第一次显示时候还是需要调用方法将其显示出来。

3.32. DUI ActiveX 控件

类名：CDuiActiveX

控件名：activex

说明：ActiveX 控件，是从 CControlBase 派生的，所以基类控件具有的属性和函数也可以使用。

属性：

属性名	类型	说明
clsid	字符串	ActiveX 控件的 CLSID
url	字符串	导航 URL
delaycreate	0|1	是否延迟创建 ActiveX 控件，直到控件需要显示的时候才创建
show-contextmenu	0|1	是否显示控件自己的右键菜单，默认是显示
show-scroll	0|1	是否显示控件自己的滚动条，默认是显示

函数：

函数	是否虚函数	说明
CreateControl	否	创建 ActiveX 控件
ParseFilePath	否	解析传入的文件路径

3.33. Web 浏览器控件(IE 内核)

类名：CDuiWebBrowser

控件名：webbrowser

说明：浏览器控件，是从 CDuiActiveX 派生的，所以基类控件具有的属性和函数也可以使用。

属性：

属性名	类型	说明
url	字符串	导航 URL，如果以 file://开头，则表示加载本地文件，后面跟的是本地文件的地址，本地文件支持相对路径，如果没有:，则表示文件以当前 exe 文件的路径作为相对路径
delaycreate	0|1	是否延迟创建浏览器控件，直到控件需要显示的时候才创建
show-contextmenu	0|1	是否显示控件自己的右键菜单
show-scroll	0|1	是否显示控件自己的滚动条，默认是显示
duimsg	字符串	指定控件会发送哪些 DUI 消息，消息名之间用|分隔，例如：” msg1|msg2|msg3”。 此控件支持的消息名包括： invoke – 浏览器的 Invoke 消息转发给 DUI 消息

函数：

函数	是否虚函数	说明
Navigate	是	浏览器导航
GoBack	否	导航到上一个 URL
GoForward	否	导航到下一个 URL
Refresh	否	刷新页面
Stop	否	停止加载页面

事件说明：

Webbrowser 控件可以截获浏览器的调用事件，并将事件通过 DUI 消息发送给事件处理类进行处理，只需要按照事件处理类的定义方法进行定义和实现就可以，事件处理类中 Msg 参数对应的浏览器 Invoke 函数的 dispidMember 参数，wParam 参数对应的 Invoke 函数的 pdispparams 参数，lParam 参数对应的 Invoke 函数的 pvarResult，浏览器 Invoke 函数原型如下：

Invoke(DISPID dispidMember,REFIID riid,LCID lcid,WORD wFlags,DISPPARAMS FAR* pdispparams, VARIANT FAR* pvarResult,EXCEPINFO FAR* pexcepinfo,UINT FAR* puArgErr);

常用的页面加载进度、文件下载情况、URL 导航前、导航后、页面加载完成、新建窗口等消息都可以收到。

界面示例：

3.34. wke 浏览器控件(webkit 内核)

类名：CDuiWkeView

控件名：wkeview

说明：webkit 内核的浏览器控件，是从 CControlBase 派生的，所以基类控件具有的属性和函数也可以使用。

Wke 是一个开源的 webkit 浏览器内核的封装，特点是体积比较小，动态库只有 10M 左右，wke 精简掉了一些桌面 UI 不常用的功能，例如音频、视频有关的功能，如果你需要播放音频或视频，可以在网页中嵌入 flash。

Wke 并不提供一个完整的浏览器所具有的一些交互功能，例如 alert，上传文件对话框，右键菜单等等，其中 alert 可以通过注册外部的实现函数来提供。

wke 会在 EXE 目录下加载 plugins 目录下的插件，并且在注册表中自动搜索 HKCU\Software\MozillaPlugins 下面注册的插件路径并加载。web.kit 扩展库已经默认包含了 flash 插件，但这个插件不是必须的，如果不使用 flash，你可以在发布后删掉 "\plugins\NPSWF32.dll" 这个文件。

属性：

属性名	类型	说明
url	字符串	导航 URL，如果以 file://开头，则表示加载本地文件，后面跟的是本地文件的地址，本地文件支持相对路径，如果没有:，则表示文件以当前 exe 文件的路径作为相对路径
html	字符串	Html 页面内容 ,如果没有设置 url 属性 ,设置 html 属性才会在控件创建时候加载 html 内容，否则会优先加载 url 属性设置的地址
delaycreate	0|1	是否延迟创建浏览器控件，直到控件需要显示的时候才创建
transparent	0|1	是否设置为透明背景，可用于使用 wke 控件制作异形窗口
duimsg	字符串	指定控件会发送哪些 DUI 消息，消息名之间用|分隔，例如：" msg1|msg2|msg3" 。 此控件支持的消息名包括：

函数：

函数	是否虚函数	说明
Navigate	是	浏览器导航
canGoBack	否	是否可以导航到上一个 URL
goBack	否	导航到上一个 URL
canGoForward	否	是否可以导航到下一个 URL
goForward	否	导航到下一个 URL

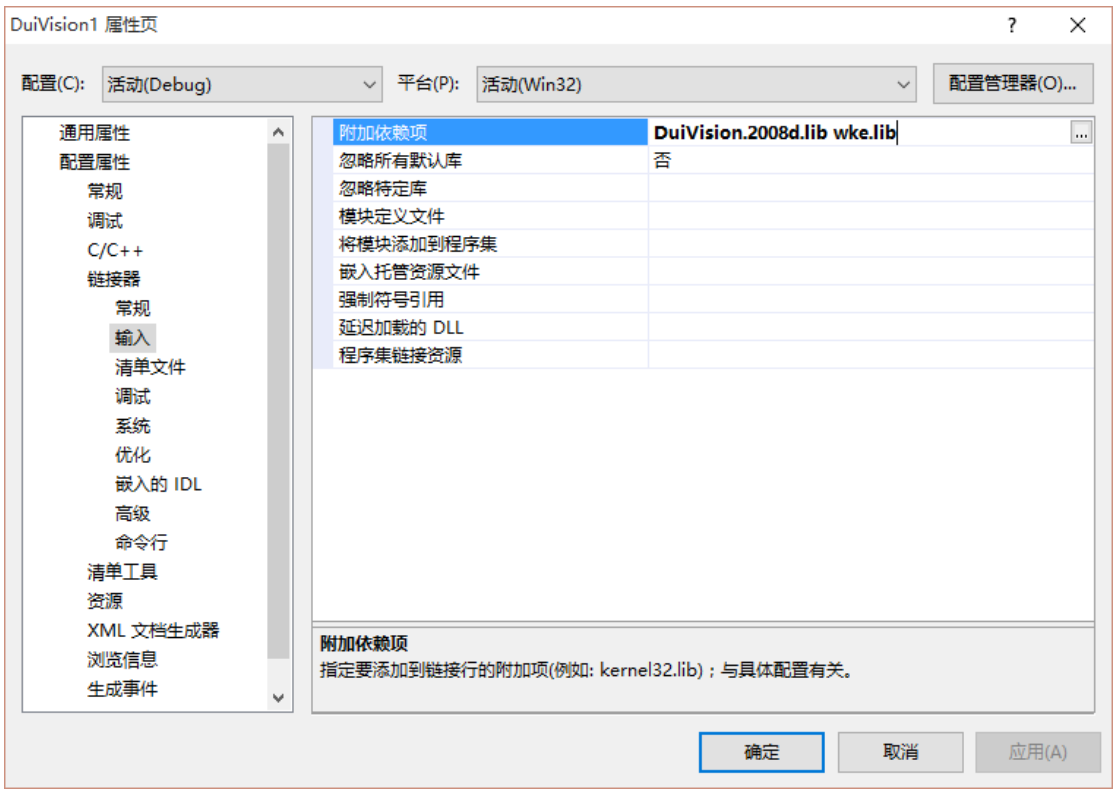
事件说明：

注意事项：

1、如果程序中使用了 WKE 控件，就需要依赖 wke.dll 动态库，这个文件比较大，有 11M，如果不需要使用 WKE 控件，因此 DuiVision 向导创建的工程默认没有包含 wke 的支持，如果要在程序中支持 Wke 控件，有两种方法：

1、使用向导创建 DuiVision 工程时候，勾选“支持 WebKit 浏览器控件”的选项，这样生成的工程会添加 Wke 的 lib 文件配置、拷贝 Wke.dll 文件、添加 Wke 控件的注册代码；

2、如果创建工程时候没有勾选此选项，也可以人工在工程中添加相关的配置和代码，添加的配置内容在工程属性的”链接器->输入”页面，附加依赖项增加 wke.lib:



然后在工程的主程序 cpp 文件中增加如下的头文件引用：

```
#include "DuiWkeView.h"
```

在主程序 cpp 文件的 InitInstance 中 DuiSystem 初始化之后，增加下面蓝色部分的代码，用于注册 Wke 控件：

```
BOOL CDuiVision1App::InitInstance()
{
	CWinApp::InitInstance();

	AfxEnableControlContainer();
```

```
	// TODO: 应适当修改该字符串，
	// 例如修改为公司或组织名
	SetRegistryKey(_T("DuiVision1"));

	// 初始化DuiVision界面库,可以指定语言,dwLangID为表示自动判断当前语言
	// 11160是应用程序ID,每个DUI应用程序应该使用不同的ID,ID主要用于进程间通信传递命令行时候区分应用
	DWORD dwLangID = 0;
	new DuiSystem(m_hInstance, dwLangID, _T("DuiVision1.ui"), 11160, IDD_DUIVISIONAPP_DIALOG, _T(""));

	// 注册WKE控件
	REGISTER_DUICONTROL(CDuiWkeView, CDuiWkeView::WkeShutdown);

	// 创建主窗口
	CDlgBase* pMainDlg = DuiSystem::CreateDuiDialog(_T("dlg_main"), NULL, _T(""), TRUE);
	// 给主窗口注册事件处理对象
	CDuiHandlerMain* pHandler = new CDuiHandlerMain();
	pHandler->SetDialog(pMainDlg);
	DuiSystem::RegisterHandler(pMainDlg, pHandler);

	// 初始化提示信息窗口
	DuiSystem::Instance()->CreateNotifyMsgBox(_T("dlg_notifymsg"));

	// 按照非模式对话框创建主窗口,可以默认隐藏
	pMainDlg->Create(pMainDlg->GetIDTemplate(), NULL);
	INT_PTR nResponse = pMainDlg->RunModalLoop();

	// 释放DuiVision界面库的资源
	DuiSystem::Release();

	// 由于对话框已关闭，所以将返回FALSE 以便退出应用程序，
	//  而不是启动应用程序的消息泵。
	return FALSE;
}
```

3.35. Flash 控件

类名：CDuiFlash

控件名：flash

说明：Flash 控件，是从 CDuiActiveX 派生的，所以基类控件具有的属性和函数也可以使用。

属性：

属性名	类型	说明
url	字符串	导航 URL
delaycreate	0|1	是否延迟创建 ActiveX 控件，直到控件需要显示的时候才创建
show-contextmenu	0|1	是否显示控件自己的右键菜单
transparent	数字	透明度，0-100 的数字
vars	字符串	指定 Flash 控件的参数，例如通过 flash 播放多媒体文件，通过参数 controlbar=over 设置显示播放器的控制条

函数：

函数	是否虚函数	说明
Navigate	否	浏览器导航
PutFlashVars	否	设置 flash 参数
isExistFlashActiveX	否	检测是否安装了 flash 控件，是一个静态函数

界面示例：

3.36. 媒体播放器控件

类名：CDuiMediaPlayer

控件名：mediaplayer

说明：媒体播放器控件，是从 CDuiActiveX 派生的，所以基类控件具有的属性和函数也可以使用。

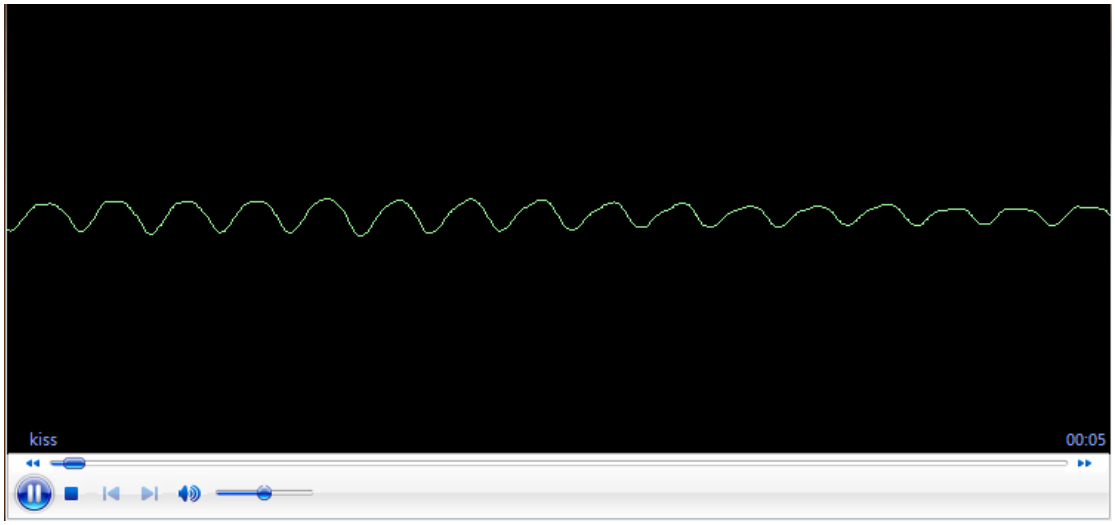
属性：

属性名	类型	说明
url	字符串	导航 URL
delaycreate	0|1	是否延迟创建 ActiveX 控件，直到控件需要显示的时候才创建
show-contextmenu	0|1	是否显示控件自己的右键菜单
transparent	数字	透明度，0-100 的数字

函数：

函数	是否虚函数	说明
Navigate	否	浏览器导航

界面示例：


kiss
00:05