从Bkwin到DUIEngine快速入门

DuiEngine是一个基于金山Bkwin修改完善形成的一个新界面库，该库吸取了Bkwin的精华，经过全面重构对原有不合理或缺失的地方进行修正。本文主要是指导原使用bkwin 的人员快速入门。

一、资源处理方式上的区别

入门的第一步就是了解DUIEngine的资源处理方式，搞清楚这个后面的相对就比较easy了。原来熟悉bkwin的同学，都知道bkwin的资源是这么干的：

|  |  |  |
| --- | --- | --- |
| **文件** | **说明** | **例子** |
| **bkwinres.h** | **控件ID的宏定义** | **#define IDC\_BTN\_ CLOSE 110001** |
| **bkwinres.rc2** | **资源定义** | **定义XML,PNG,BMP等资源**  **DEFINE\_XML(IDR\_SKIN\_DEF,"res\\app\_skin.xml")** |
| app\_string.xml | 定义字符串 | <s id="10">您好</s> |
| app\_skin.xml | 定义皮肤 | <png name="SysBtnClose" src="22001" subwidth="39"/> |
| app\_style.xml | 定义风格 | <class name="TitleText" font="0004" crtext="FFFFFF"/> |
| **App\_main.xml** | **布局定义** | **定义窗口的而已** |

* **index.xml和residbuilder工具**

工欲善其事,必先利其器。在DUIEngine里，引入了一个index.xml和residbuilder工具。Index.xml其实和原来bkwinres.rc2的作用是一样的，就是来定义资源。

<resid type="XML" id\_name="IDR\_DUI\_SKIN\_DEF" id="100" file="layout\def\_skin.xml"/>

<resid type="XML" id\_name="IDR\_DUI\_STYLE\_DEF" id="101" file="layout\def\_style.xml" />

<resid type="XML" id\_name="IDR\_DUI\_STRING\_DEF" id="102" file="layout\def\_string.xml" />

这个只是指定了资源类型、资源名称、资源ID。需要经过residbuilder工具来生成以下文件：

Duires\winres.rc2： 是根据index.xml动态生成的，就是以前的bkwinres.rc2。

Winres.h： 自动根据布局文件（如dlg\_mail.xml）生成的控件的宏定义。在布局文件里，每个控件多了个name，residbuilder就是根据name和id生成出来宏定义。另外，还可以自动为编出id。

name2id.xml：这个自动生成出来的NAME和ID的映射关系，主要是程序内部使用。

Residbuider的使用说明：

residbuilder -y -i .\skin\index.xml -r .\duires\winres.rc2 -n .\duires\name2id.xml -h .\duires\winres.h

-y: 强制改写(只读文件)

-i: 资源定义文件 \*所有用到的资源需要在此处定义

-r: 生成的资源rc2文件 \*将资源加入PE文件

-h: 生成的头文件 \*生成资源名称和ID的宏定义

-n: 生成的名称ID映射文件 \*生成名称和ID的对应关系

-p: 皮肤路径 \*相对于程序的.rc文件

* **窗口布局定义**

和bkwin相比，DUIEngine抛弃了原来的Head、Body、Foot的三部分做法，只剩一个Body了。

<layer **title="DUI-DEMO"** width="600" height="400" appwin="1" **ncRect="5,5,5,5"** resize="1" **translucent="0"** nomaxsize="0">

<body>

<dlg **name**="maindlg" pos="0,0,-0,-0" class="cls\_dlg\_frame">

**<caption pos="0,0,-0,29">**

<text pos="11,9" class="cls\_txt\_red">DUI-Demo ver:%str1%</text>

<imgbtn id="1" name="close" skin="btn\_sys\_close" pos="-45,0" tip="close" animate="0"/>

**</caption>**

</dlg>

</body>

</layer>

红色的部分就是原来新增的或变化比较大的。在DuiEngine里，定义skin的时候可以指定属主，如：

<imgframe name="dlg\_frame" owner="DUI-DEMO" src="1" …>

<layer **title="DUI-DEMO" …**> 因是指定一个属主，每次在打一个窗口的时候，都会加载这个窗口的私有资源，比如一般很少用的对话框，以减少资源的使用。

**…ncRect="5,5,5,5"…** 就是指定窗口的非客户区, 比如 Resize的时候，到这个非客户区就会出现光标指针改变。

**…translucent…** 指定窗口半透明。

**<caption …>** 新增，标题栏。在标题栏双击可最大化、还原窗口，还可以拖动窗口。

* **常用控件的属性定义**

def\_objattr.xml。 针对经常使用控件的不变属性，在此处定义，以免每次都要写。

<button class="normalbtn"/>

<check skin="btncheckbox"/>

<radio skin="btnRadio"/>

* **资源的几种加载方式**
* 从ZIP文件加载皮肤

DuiResProviderZip \*zipSkin=new DuiResProviderZip;

CString strZip=CA2T(szCurrentDir)+\_T("\\def\_skin.zip");

if(!zipSkin->Init(strZip))

{

DUIASSERT(0);

return 1;

}

duiSystem.SetResProvider(zipSkin);

* 从文件加载皮肤

lstrcatA( szCurrentDir, "\\..\\dui\_demo" );

DuiResProviderFiles \*pResFiles=new DuiResProviderFiles;

if(!pResFiles->Init(szCurrentDir))

{

DUIASSERT(0);

return 1;

}

duiSystem.SetResProvider(pResFiles);

* 从资源加载皮肤

duiSystem.SetResProvider(new DuiResProviderPE(hInstance));

注意：对于第三种，需要将winres.rc2加入到资源里。

二、类图

