DirectUI是通过xml文件来描述界面的，DirectUI库通过解析xml文件来呈现界面，从而实现界面与逻辑的分离。当然，xml需要遵照特定的规矩描述界面元素的名称、坐标、大小、配色方案、背景图片等多种属性。下面我们给出一个xml文件的实例：

<?xml version="1.0" encoding="UTF-8"?>

<Window size="500,350" caption="0,0,0,36" roundcorner="4,4" >

<Default name="VScrollBar" value="button1normalimage=&quot;file='scrollbar.bmp' source='0,90,16,106' mask='#FFFF00FF'&quot; button1hotimage=&quot;file='scrollbar.bmp' source='18,90,34,106' mask='#FFFF00FF'&quot; button1pushedimage=&quot;file='scrollbar.bmp' source='36,90,52,106' mask='#FFFF00FF'&quot; button1disabledimage=&quot;file='scrollbar.bmp' source='54,90,70,106' mask='#FFFF00FF'&quot; button2normalimage=&quot;file='scrollbar.bmp' source='0,108,16,124' mask='#FFFF00FF'&quot; button2hotimage=&quot;file='scrollbar.bmp' source='18,108,34,124' mask='#FFFF00FF'&quot; button2pushedimage=&quot;file='scrollbar.bmp' source='36,108,52,124' mask='#FFFF00FF'&quot; button2disabledimage=&quot;file='scrollbar.bmp' source='54,108,70,124' mask='#FFFF00FF'&quot; thumbnormalimage=&quot;file='scrollbar.bmp' source='0,126,16,142' corner='2,2,2,2' mask='#FFFF00FF'&quot; thumbhotimage=&quot;file='scrollbar.bmp' source='18,126,34,142' corner='2,2,2,2' mask='#FFFF00FF'&quot; thumbpushedimage=&quot;file='scrollbar.bmp' source='36,126,52,142' corner='2,2,2,2' mask='#FFFF00FF'&quot; thumbdisabledimage=&quot;file='scrollbar.bmp' source='54,126,70,142' corner='2,2,2,2' mask='#FFFF00FF'&quot; railnormalimage=&quot;file='scrollbar.bmp' source='0,144,16,160' corner='2,2,2,2' mask='#FFFF00FF'&quot; railhotimage=&quot;file='scrollbar.bmp' source='18,144,34,160' corner='2,2,2,2' mask='#FFFF00FF'&quot; railpushedimage=&quot;file='scrollbar.bmp' source='36,144,52,160' corner='2,2,2,2' mask='#FFFF00FF'&quot; raildisabledimage=&quot;file='scrollbar.bmp' source='54,144,70,160' corner='2,2,2,2' mask='#FFFF00FF'&quot; bknormalimage=&quot;file='scrollbar.bmp' source='0,162,16,178' corner='2,2,2,2' mask='#FFFF00FF'&quot; bkhotimage=&quot;file='scrollbar.bmp' source='18,162,34,178' corner='2,2,2,2' mask='#FFFF00FF'&quot; bkpushedimage=&quot;file='scrollbar.bmp' source='36,162,52,178' corner='2,2,2,2' mask='#FFFF00FF'&quot; bkdisabledimage=&quot;file='scrollbar.bmp' source='54,162,70,178' corner='2,2,2,2' mask='#FFFF00FF'&quot; " />

<VerticalLayout inset="4,0,4,4" bkimage="file='winbk.bmp' corner='4,48,4,4' mask='#FFFF00FF'" >

<HorizontalLayout height="32" inset="2,2,2,0">

<Label text="PCLintAssist" width="120" />

<Button name="CloseBtn" width="42" maxheight="18" tooltip="点击这里直接退出" normalimage="file='frame\_btn\_close\_normal.bmp' mask='#FFFF00FF'" hotimage="file='frame\_btn\_close\_hot.bmp' mask='#FFFF00FF'" pushedimage="file='frame\_btn\_close\_down.bmp' mask='#FFFF00FF'"/>

</HorizontalLayout>

<Container bkcolor="#FFFFFFFF" >

<Label text="工程文件：" float="true" pos="10, 20, 78, 42" />

<Edit name="PrjFileEdit" float="true" pos="80, 20, 420, 42" password="false" tooltip="请在这里输入工程文件全路径" bordercolor="#FF4EA0D1" bordersize="1"/>

<Button name="BrowsePrjFileBtn" text="Browse" float="true" pos="425, 20,485, 42" tooltip="选择工程文件路径" normalimage="button\_nor.bmp" hotimage="button\_over.bmp" pushedimage="button\_down.bmp" />

<Label text="PCLint目录：" float="true" pos="10, 55, 78, 77" />

<Edit name="LintFileEdit" float="true" pos="80, 55, 420, 77" password="false" tooltip="在这里输入PClint的安装目录" bordercolor="#FF4EA0D1" bordersize="1"/>

<Button name="BrowseLintBtn" text="Browse" float="true" pos="425, 55,485, 77" tooltip="选择PCLint安装目录" normalimage="button\_nor.bmp" hotimage="button\_over.bmp" pushedimage="button\_down.bmp" />

<Label text="头文件目录：" float="true" pos="10, 90, 78, 112" />

<Edit name="IncFileEdit" float="true" pos="80, 90, 420, 112" password="false" tooltip="在这里输入头文件目录，可多个，以;隔开" bordercolor="#FF4EA0D1" bordersize="1"/>

<Button name="BrowseIncBtn" text="Browse" float="true" pos="425, 90,485, 112" tooltip="选择头文件目录，可多次选择" normalimage="button\_nor.bmp" hotimage="button\_over.bmp" pushedimage="button\_down.bmp" />

<Option text="" pos="10,125,78,147" float="true" normalimage="file='navigationbar.bmp' source='0,0,18,18'" selectedimage="file='navigationbar.bmp' source='18,0,36,18'" group="true"/>

<Label text="链接 Visual studio include file" float="true" pos="10, 120, 78, 300" />

<Button name="OKBtn" text="OK" float="true" pos="220, 250, 280, 272" tooltip="Make file." normalimage="button\_nor.bmp" hotimage="button\_over.bmp" pushedimage="button\_down.bmp" />

<Label name="tips" text="" float="true" pos="10, 285, 250, 607" />

</Container>

</VerticalLayout>

</Window>

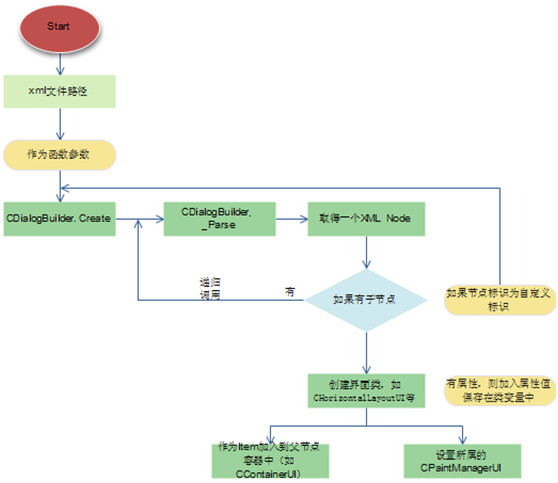
这个xml经过解析渲染后，呈现的界面如图所示：



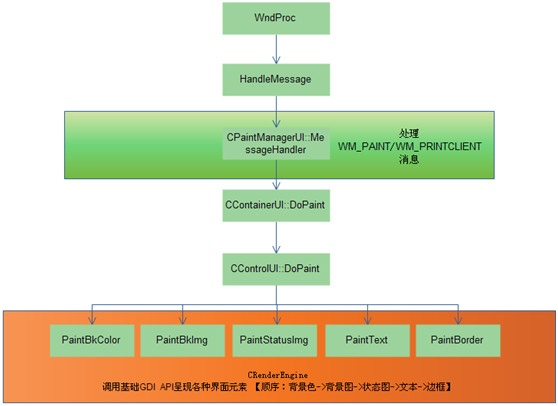
界面效果还可以把？！那么，DirectUI是如何做到这点的呢？DirectUI呈现界面的核心类库是CDialogBuilder/CPaintManagerUI/CRenderEngine。

其中，CDialogBuilder负责遍历xml节点，获取界面元素并构建响应的类；CPaintManagerUI负责消息分发，是整个DirectUI工作的基础；而CRenderEngine则负责具体调用GDI API完成具体的界面渲染工作。

解析XML文件生成响应界面类过程可以使用下面的流程图形象表示：



解析xml并完成界面元素类的构建后，就需要对这些界面元素进行渲染显示了。说道渲染，在windows消息体系来看，主要是对WM\_PAINT以及WM\_PRINTCLIENT等消息的处理了。下面的流程图描述了这个渲染处理过程：



为更好的理解这个流程，下面贴出代码片段：

传递xml文件路径并解析：

CControlUI\* CDialogBuilder::Create(IDialogBuilderCallback\* pCallback, CPaintManagerUI\* pManager, CControlUI\* pParent)

{

.....

return \_Parse(&root, pParent, pManager);

}

遍历xml节点，构建界面元素相关类：

CControlUI\* CDialogBuilder::\_Parse(CMarkupNode\* pRoot, CControlUI\* pParent, CPaintManagerUI\* pManager)

{

IContainerUI\* pContainer = NULL;

CControlUI\* pReturn = NULL;

for( CMarkupNode node = pRoot->GetChild() ; node.IsValid(); node = node.GetSibling() ) {

LPCTSTR pstrClass = node.GetName();

if( \_tcscmp(pstrClass, \_T("Image")) == 0 || \_tcscmp(pstrClass, \_T("Font")) == 0 \

|| \_tcscmp(pstrClass, \_T("Default")) == 0 ) continue;

CControlUI\* pControl = NULL;

if( \_tcscmp(pstrClass, \_T("Include")) == 0 ) {

if( !node.HasAttributes() ) continue;

int count = 1;

LPTSTR pstr = NULL;

TCHAR szValue[500] = { 0 };

SIZE\_T cchLen = lengthof(szValue) - 1;

if ( node.GetAttributeValue(\_T("count"), szValue, cchLen) )

count = \_tcstol(szValue, &pstr, 10);

cchLen = lengthof(szValue) - 1;

if ( !node.GetAttributeValue(\_T("source"), szValue, cchLen) ) continue;

for ( int i = 0; i < count; i++ ) {

CDialogBuilder builder;

if( m\_pstrtype != NULL ) { // 使用资源dll，从资源中读取

WORD id = (WORD)\_tcstol(szValue, &pstr, 10);

pControl = builder.Create((UINT)id, m\_pstrtype, m\_pCallback, pManager, pParent);

}

else {

pControl = builder.Create((LPCTSTR)szValue, (UINT)0, m\_pCallback, pManager, pParent);

}

}

continue;

}

else {

SIZE\_T cchLen = \_tcslen(pstrClass);

switch( cchLen ) {

case 4:

if( \_tcscmp(pstrClass, \_T("Edit")) == 0 ) pControl = new CEditUI;

else if( \_tcscmp(pstrClass, \_T("List")) == 0 ) pControl = new CListUI;

else if( \_tcscmp(pstrClass, \_T("Text")) == 0 ) pControl = new CTextUI;

break;

case 5:

if( \_tcscmp(pstrClass, \_T("Combo")) == 0 ) pControl = new CComboUI;

else if( \_tcscmp(pstrClass, \_T("Label")) == 0 ) pControl = new CLabelUI;

break;

case 6:

if( \_tcscmp(pstrClass, \_T("Button")) == 0 ) pControl = new CButtonUI;

else if( \_tcscmp(pstrClass, \_T("Option")) == 0 ) pControl = new COptionUI;

else if( \_tcscmp(pstrClass, \_T("Slider")) == 0 ) pControl = new CSliderUI;

break;

case 7:

if( \_tcscmp(pstrClass, \_T("Control")) == 0 ) pControl = new CControlUI;

else if( \_tcscmp(pstrClass, \_T("ActiveX")) == 0 ) pControl = new CActiveXUI;

break;

case 8:

if( \_tcscmp(pstrClass, \_T("Progress")) == 0 ) pControl = new CProgressUI;

else if( \_tcscmp(pstrClass, \_T("RichEdit")) == 0 ) pControl = new CRichEditUI;

// add by:zjie

else if (\_tcscmp(pstrClass, \_T("CheckBox")) == 0) pControl = new CCheckBoxUI;

else if (\_tcscmp(pstrClass, \_T("ComboBox")) == 0) pControl = new CComboBoxUI;

else if (\_tcscmp(pstrClass, \_T("DateTime")) == 0) pControl = new CDateTimeUI;

// add by:zjie

break;

case 9:

if( \_tcscmp(pstrClass, \_T("Container")) == 0 ) pControl = new CContainerUI;

else if( \_tcscmp(pstrClass, \_T("TabLayout")) == 0 ) pControl = new CTabLayoutUI;

else if( \_tcscmp(pstrClass, \_T("ScrollBar")) == 0 ) pControl = new CScrollBarUI;

break;

case 10:

if( \_tcscmp(pstrClass, \_T("ListHeader")) == 0 ) pControl = new CListHeaderUI;

else if( \_tcscmp(pstrClass, \_T("TileLayout")) == 0 ) pControl = new CTileLayoutUI;

else if( \_tcscmp(pstrClass, \_T("WebBrowser")) == 0 ) pControl = new CWebBrowserUI;

break;

case 11:

if (\_tcscmp(pstrClass, \_T("ChildWindow")) == 0) pControl=new CChildWindowUI;

break;

case 14:

if( \_tcscmp(pstrClass, \_T("VerticalLayout")) == 0 ) pControl = new CVerticalLayoutUI;

else if( \_tcscmp(pstrClass, \_T("ListHeaderItem")) == 0 ) pControl = new CListHeaderItemUI;

break;

case 15:

if( \_tcscmp(pstrClass, \_T("ListTextElement")) == 0 ) pControl = new CListTextElementUI;

break;

case 16:

if( \_tcscmp(pstrClass, \_T("HorizontalLayout")) == 0 ) pControl = new CHorizontalLayoutUI;

else if( \_tcscmp(pstrClass, \_T("ListLabelElement")) == 0 ) pControl = new CListLabelElementUI;

break;

case 20:

if( \_tcscmp(pstrClass, \_T("ListContainerElement")) == 0 ) pControl = new CListContainerElementUI;

break;

}

// User-supplied control factory

if( pControl == NULL ) {

CStdPtrArray\* pPlugins = CPaintManagerUI::GetPlugins();

LPCREATECONTROL lpCreateControl = NULL;

for( int i = 0; i < pPlugins->GetSize(); ++i ) {

lpCreateControl = (LPCREATECONTROL)pPlugins->GetAt(i);

if( lpCreateControl != NULL ) {

pControl = lpCreateControl(pstrClass);

if( pControl != NULL ) break;

}

}

}

if( pControl == NULL && m\_pCallback != NULL ) {

pControl = m\_pCallback->CreateControl(pstrClass);

}

}

ASSERT(pControl);

if( pControl == NULL ) continue;

// Add children

if( node.HasChildren() ) {

\_Parse(&node, pControl, pManager);

}

// Attach to parent

// 因为某些属性和父窗口相关，比如selected，必须先Add到父窗口

if( pParent != NULL ) {

if( pContainer == NULL ) pContainer = static\_cast<IContainerUI\*>(pParent->GetInterface(\_T("IContainer")));

ASSERT(pContainer);

if( pContainer == NULL ) return NULL;

if( !pContainer->Add(pControl) ) {

delete pControl;

continue;

}

}

// Init default attributes

if( pManager ) {

pControl->SetManager(pManager, NULL, false);

LPCTSTR pDefaultAttributes = pManager->GetDefaultAttributeList(pstrClass);

if( pDefaultAttributes ) {

pControl->ApplyAttributeList(pDefaultAttributes);

}

}

// Process attributes

if( node.HasAttributes() ) {

TCHAR szValue[500] = { 0 };

SIZE\_T cchLen = lengthof(szValue) - 1;

// Set ordinary attributes

int nAttributes = node.GetAttributeCount();

for( int i = 0; i < nAttributes; i++ ) {

pControl->SetAttribute(node.GetAttributeName(i), node.GetAttributeValue(i));

}

}

if( pManager ) {

pControl->SetManager(NULL, NULL, false);

}

// Return first item

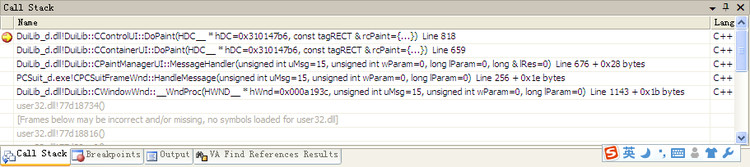
if( pReturn == NULL ) pReturn = pControl;

}

return pReturn;

}

渲染过程：



 每个界面元素类均执行DoPaint操作：

void CControlUI::DoPaint(HDC hDC, const RECT& rcPaint)

{

if( !::IntersectRect(&m\_rcPaint, &rcPaint, &m\_rcItem) ) return;

// 绘制循序：背景颜色->背景图->状态图->文本->边框

if( m\_cxyBorderRound.cx > 0 || m\_cxyBorderRound.cy > 0 ) {

CRenderClip roundClip;

CRenderClip::GenerateRoundClip(hDC, m\_rcPaint, m\_rcItem, m\_cxyBorderRound.cx, m\_cxyBorderRound.cy, roundClip);

PaintBkColor(hDC);

PaintBkImage(hDC);

PaintStatusImage(hDC);

PaintText(hDC);

PaintBorder(hDC);

}

else {

PaintBkColor(hDC);

PaintBkImage(hDC);

PaintStatusImage(hDC);

PaintText(hDC);

PaintBorder(hDC);

}

}

PaintBkColor代码，其它Paint\*\*\*过程类似：

void CControlUI::PaintBkColor(HDC hDC)

{

if( m\_dwBackColor != 0 ) {

if( m\_dwBackColor2 != 0 ) {

if( m\_dwBackColor3 != 0 ) {

RECT rc = m\_rcItem;

rc.bottom = (rc.bottom + rc.top) / 2;

CRenderEngine::DrawGradient(hDC, rc, GetAdjustColor(m\_dwBackColor), GetAdjustColor(m\_dwBackColor2), true, 8);

rc.top = rc.bottom;

rc.bottom = m\_rcItem.bottom;

CRenderEngine::DrawGradient(hDC, rc, GetAdjustColor(m\_dwBackColor2), GetAdjustColor(m\_dwBackColor3), true, 8);

}

else

CRenderEngine::DrawGradient(hDC, m\_rcItem, GetAdjustColor(m\_dwBackColor), GetAdjustColor(m\_dwBackColor2), true, 16);

}

else if( m\_dwBackColor >= 0xFF000000 ) CRenderEngine::DrawColor(hDC, m\_rcPaint, GetAdjustColor(m\_dwBackColor));

else CRenderEngine::DrawColor(hDC, m\_rcItem, GetAdjustColor(m\_dwBackColor));

}

}

看完本文，对从xml到真正的界面呈现过程是不是清晰了很多呢？