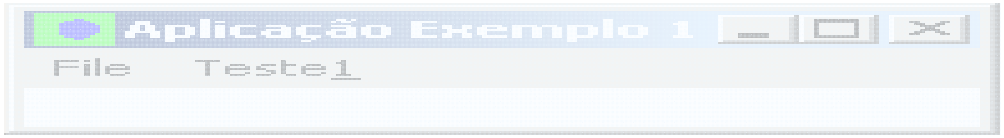


Modo GUI na API Win32 – III

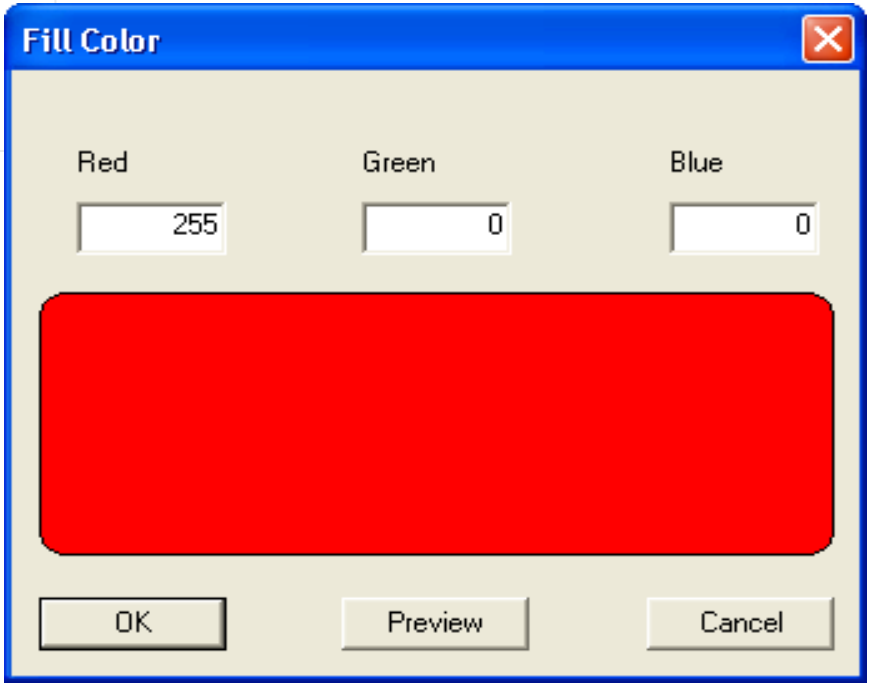
WM_PAINT *in Dialog Boxes*

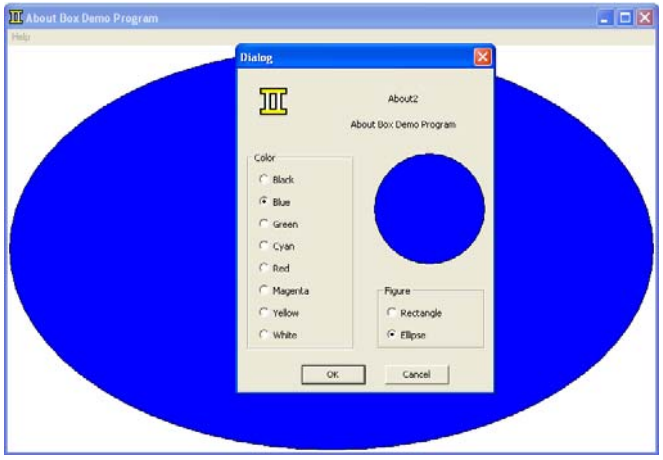
Images in Picture controls

Common controls

Desenhar sobre uma Dialog Box

- Desenhar sobre uma área de uma Dialog Box
  - Desenhar directamente sobre a Dialog Box – o que implica conhecer as coordenadas de desenho
  - Desenhar sobre um CWC (*Static control*) – o desenho é referênciado à client area do control





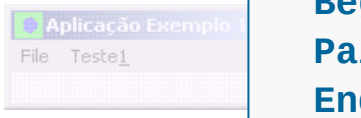
Dialog Box e WM_PAINT

- A intercepção da mensagem WM_PAINT para as janelas de diálogo é feita do mesmo modo que para as janelas “normais”:

```
LRESULT CALLBACK Dialog(HWND hDlg, UINT message, WPARAM wParam, LPARAM lParam) {
  switch ( message ) {
    ...
    HANDLE_MSG(hDlg, WM_PAINT, ClsOnPaint);
    ...
  } // end switch ( message )
}
```

Desenho sobre a face da DB


```
case IDC_ELLIPSE:
  iFigure = LOWORD (wParam) ;
  CheckRadioButton (hDlg, IDC_RECT, IDC_ELLIPSE, LOWORD(wParam);
  InvalidateRect(hDlg, NULL, TRUE);
  eturn TRUE ;
```

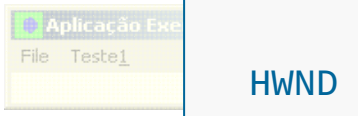



```
BOOL Cls_OnPaint(HWND hDlg) {
  BeginPaint(hDlg, &ps); // necessário para erase background *
  PaintOnTheDialogBox(hDlg, iColor, iFigure) ;
  EndPaint(hDlg, &ps);
  return FALSE;
}
```

* Se não o fizermos, o sistema vai fazê-lo no final no WM_PAINT, apagando qualquer pintura
O BeginPaint envia a mensagem de WM_ERASEBKGND se o invalidate assim o indicou.

Desenho sobre um componente

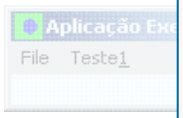

```
case IDC_ELLIPSE:
  iFigure = LOWORD (wParam) ;
  CheckRadioButton (hDlg, IDC_RECT, IDC_ELLIPSE, LOWORD (wParam)) ;

Actualização directa:
  InvalidateRect (hCtrl, NULL, TRUE) ;
  UpdateWindow (hCtrl) ; // desenhar o fundo
  PaintOnTheComponent (hCtrl, iColor, iFigure) ;

Actualização no WM_PAINT:
   InvalidateRect(hDlg, NULL, TRUE); //OK
```




```
BOOL Cls_OnPaint(HWND hDlg) {  // tem de ter sempre a capacidade de desenhar
                               // sobre o componente
  HWND hwndCtrl = GetDlgItem(hDlg,  IDC_PAINT_PREVIEW);
  UpdateWindow (hCtrlBlock); // desenhar o fundo do componente
  PaintOnTheComponent (hCtrlBlock, iColor, iFigure);  // desenhar a figura
 return FALSE;
}
```

Pintar sobre a Dialog Box

Código da função de desenho sobre a Dialog Box :

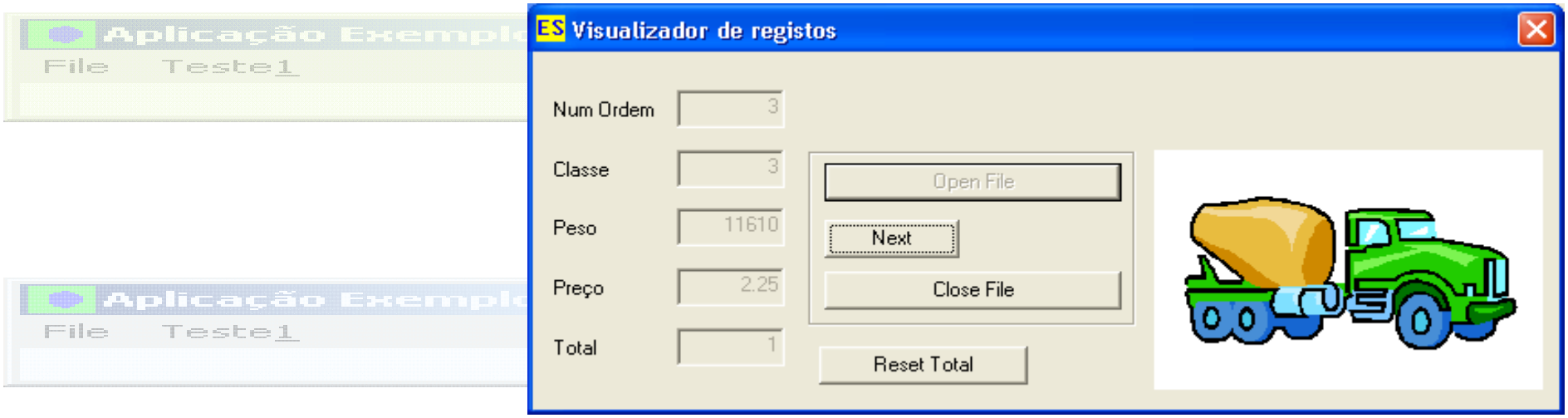



```
void PaintOnTheDialogBox(HWND hDlg ) {
   HDC hdc; HBRUSH hBrush, hOldBrush;

   hdc = GetDC (hDlg) ;
   hBrush = CreateSolidBrush (crColor[iColor - IDC_BLACK]) ;
   hOldBrush = (HBRUSH) SelectObject (hdc, hBrush) ;

   if (iFigure == IDC_RECT)
       Rectangle (hdc, 160, 100, 290, 230) ;
   else
       Ellipse   (hdc, 160, 100, 290, 230) ;

   DeleteObject (SelectObject (hdc, hOldBrush)) ;
   ReleaseDC (hDlg, hdc) ;
}
```

Pintar sobre um componente

Código da função de desenho sobre o componente :

```
void PaintOnTheComponent (HWND hwnd, int iColor, int iFigure) {
 HBRUSH hBrush ; HDC hdc; RECT rect ;

     hdc = GetDC (hwnd) ;
     GetClientRect (hwnd, &rect) ;
     hBrush = CreateSolidBrush (crColor[iColor - IDC_BLACK]) ;
     hBrush = (HBRUSH) SelectObject (hdc, hBrush) ;

     if (iFigure == IDC_RECT)
          Rectangle (hdc, rect.left+2, rect.top+2, rect.right-2, rect.bottom-2) ;
     else
          Ellipse   (hdc, rect.left+2, rect.top+2, rect.right-2, rect.bottom-2) ;

     DeleteObject (SelectObject (hdc, hBrush)) ;
     ReleaseDC (hwnd, hdc) ;
}
```

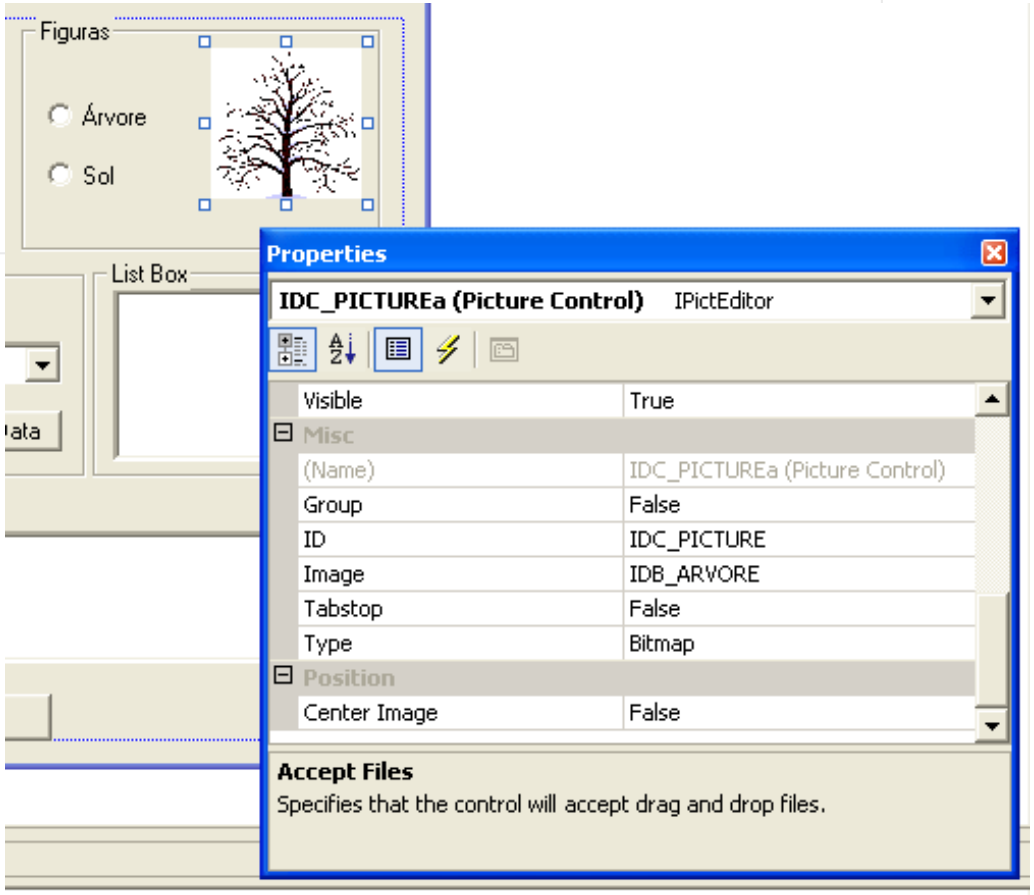


Dialog Box's

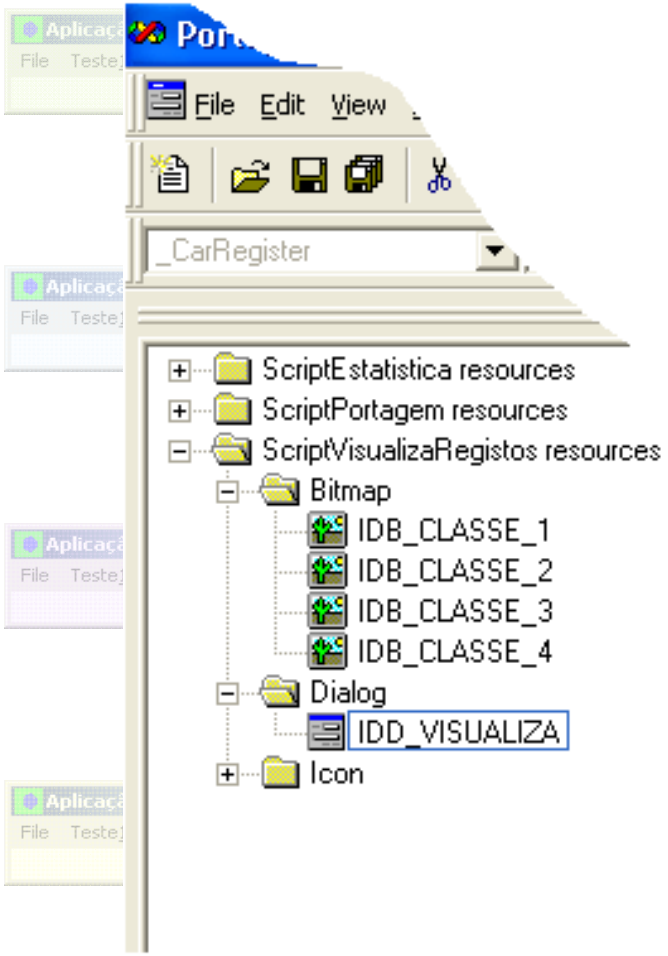
Images e Picture Control

Dialog Box e imagens

- Os *controls* do tipo Picture *Static* suportam:
  - Bitmap (.bmp),
  - Icon,
  - Rectangle,
  - ...



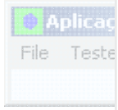
Dialog Box e imagens (cont.)



- As imagens podem estar contidas no *resource* do projecto ou podem ser carregadas de um ficheiro

Dialog Box e imagens (cont.)

- Dinamicamente é possível modificar a imagem contida no *control*
- Isso é feito com o envio da mensagem STM_SETIMAGE para o *control* onde se deseja carregar a imagem



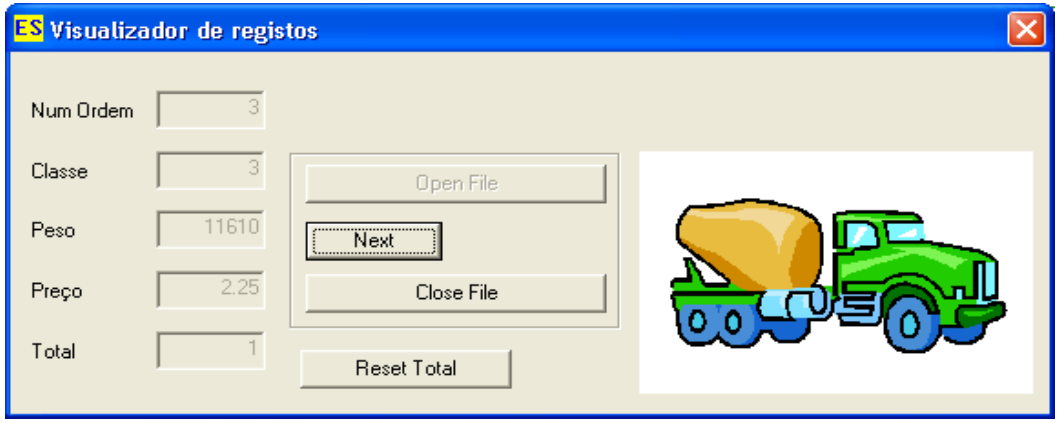

```
SendMessage(
        GetDlgItem(hDlg, IDC_ANIMACAO),
        STM_SETIMAGE,
        (WPARAM)IMAGE_BITMAP,
        (LPARAM)imagem );
```

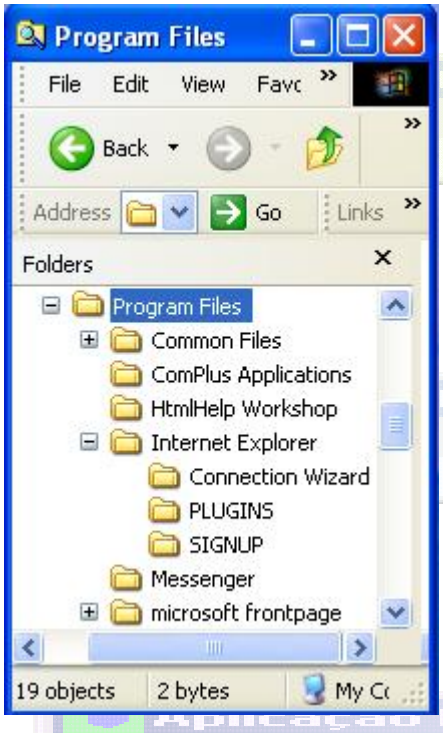
HANDLE da imagem que se pretende visualizar, o HANDLE foi previamente obtido com a função LoadImage

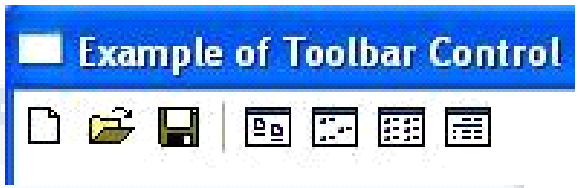
Dialog Box e imagens (ex.)

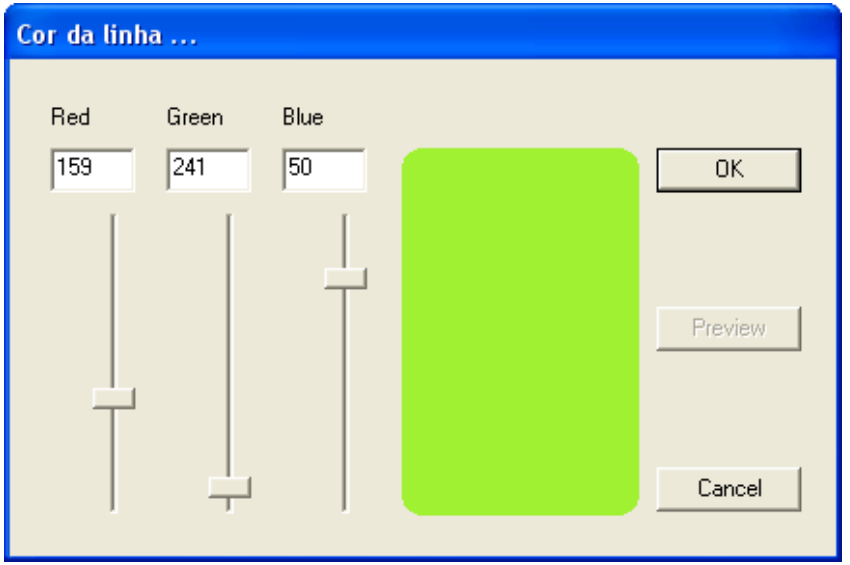
```
HANDLE himage;
//para load como resource
himage = LoadImage( GetWindowInstance(hDlg),
                    MAKEINTRESOURCE( IDB_TRUCK ),IMAGE_BITMAP, 0,0,
                    LR_DEFAULTSIZE | LR_SHARED);
// para load directamente de ficheiro
//himage = LoadImage((HINSTANCE)GetWindowInstance(hDlg),
//                    TEXT("Truck.bmp"), IMAGE_BITMAP, 0, 0,
//                    LR_DEFAULTSIZE | LR_SHARED | LR_LOADFROMFILE);

SendMessage( GetDlgItem(hDlg, IDC_ANIMACAO), STM_SETIMAGE,
              (WPARAM)IMAGE_BITMAP, (LPARAM) himage );
```

```
É uma boa oportunidade para desenvolver o
seu primeiro message craker

#define Picture_SetImage(hwndCtl, himg) ...
```



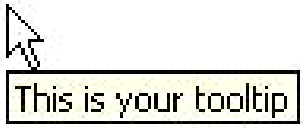


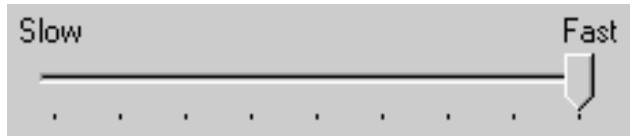


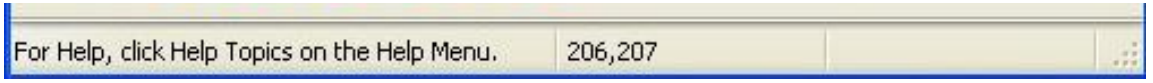


Dialog Box's

Common Controls





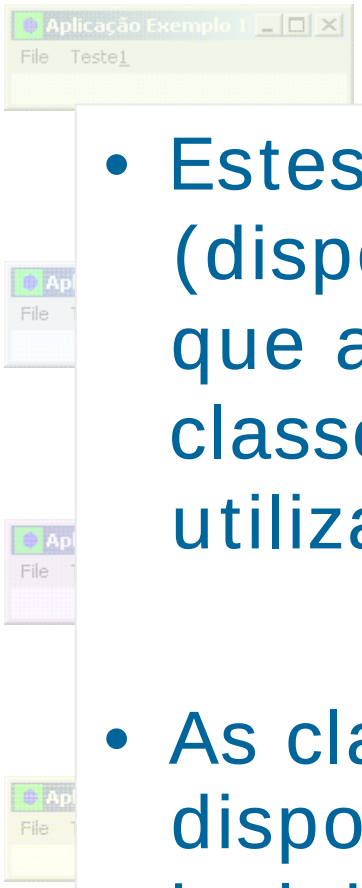


Dialog Box's e Common Controls

- Na construção de janelas de diálogo é possível utilizar um conjunto de controls com "funcionalidades extras" os designados *Common Controls*:
  - Tool Bar, Status Bar, Track Bar, Progress Bar
  - ToolTip
  - ComboBoxEx (combo boxes com imagens)
  - tree-view
  - e outros

ms-help://MS.MSDNQTR.2005JAN.1033/**shellcc/platform/commctls/wincontrols.htm**

Dialog Box's e Common Controls (cont.)

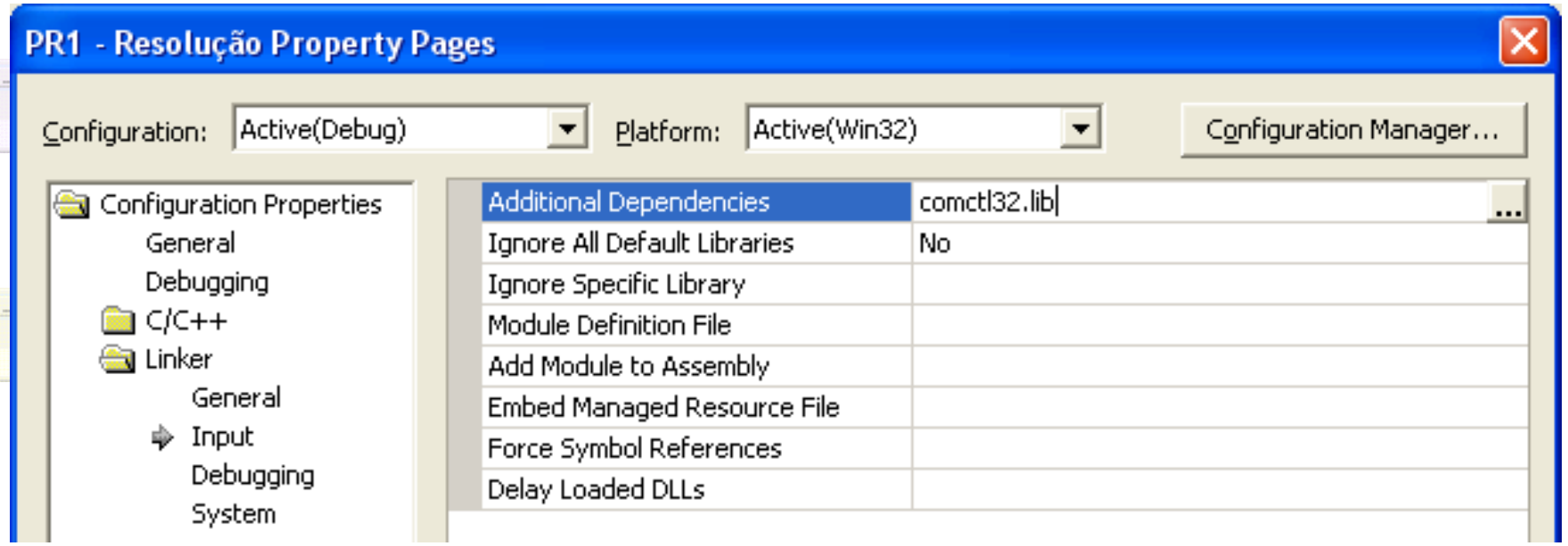


- Estes *controls* estão disponíveis em classes (disponibilizadas através de uma DLL) pelo que a sua utilização requer que a respectiva classe seja carregada antes de se poder utilizar o *control*

- As classes dos *Common Controls* estão disponíveis na DLL `ComCtl32` e é necessário incluir o ficheiro de *include* "`commctrl.h`"

Dialog Box's e Common Controls (cont.)

- A nível do projecto do *Visual Studio* é necessário adicionar a biblioteca "`Comctl32.lib`" ao projecto, isto é feito nas definições do projecto: *Project → Properties*



Dialog Box's e Common Controls (cont.)

- Na aplicação antes de usar o *control* pretendido é necessário carregar a classe do *control*. Isso é feito com a função "`InitCommonControlsEx`"

- Um exemplo para a barra de progresso é o seguinte:

```
INITCOMMONCONTROLSEX initCommonCtrl = {
        sizeof(INITCOMMONCONTROLSEX),
        ICC_BAR_CLASSES
};

InitCommonControlsEx(&initCommonCtrl);
```

Dialog Box's e Common Controls (cont.)

ICC_ANIMATE_CLASS, Load animate control class.
ICC_BAR_CLASSES, Load toolbar, status bar, trackbar, and ToolTip control classes.
ICC_COOL_CLASSES, Load rebar control class.
ICC_DATE_CLASSES, Load date and time picker control class.
ICC_HOTKEY_CLASS, Load hot key control class.
ICC_INTERNET_CLASSES, Load IP address class.
ICC_LINK_CLASS, Load a hyperlink control class.
ICC_LISTVIEW_CLASSES, Load list-view and header control classes.
ICC_NATIVEFNTCTL_CLASS, Load a native font control class.
ICC_PAGESCROLLER_CLASS, Load pager control class.
ICC_PROGRESS_CLASS, Load progress bar control class.
ICC_STANDARD_CLASSES, Load one of the intrinsic User32 control classes.
The user controls include button, edit, static, listbox, combobox, and scrollbar.
ICC_TAB_CLASSES, Load tab and ToolTip control classes.
ICC_TREEVIEW_CLASSES, Load tree-view and ToolTip control classes.
ICC_UPDOWN_CLASS, Load up-down control class.
ICC_USEREX_CLASSES, Load ComboBoxEx class.