Progress Dialog Component: Overview

The Progress Dialog component adds a progress dialog to your application. A progress dialog is what you commonly see during setup operations. It contains a progress indicator for updating the user on the completion status of a lengthy operation. The inserted progress dialog is derived from **CDialog**, and can be used in any MFC application. The progress dialog includes the following features:

- Member functions for programmatically updating the progress bar on the dialog.
- An optional **Cancel** button, along with a message handler for the button.
- An optional status text window.
- A percentage display of the current status.

Progress Dialog Component: Specifics

Progress Dialog Component: Results

Progress Dialog Component: Specifics

The progress dialog created by this component contains a Windows common progress bar control. The `SetRange`, `SetStep`, `SetPos`, `OffsetPos`, and `StepIt` member functions in the dialog class are implemented by calling the corresponding functions in the MFC class **CProgressCtrl**. For information on using these functions, see **CProgressCtrl** in the *Microsoft Foundation Class Library Reference* of the *Visual C++ Programmer's Guide*.

Furthermore, since the progress dialog class is derived from **CDialog**, and **CDialog** is derived from **CWnd**, you can use any of the functions defined in these classes as well (for example, you can use **CWnd::SetWindowText** to change the caption).

Note With the Window common progress bar control, the current position wraps back to the beginning once the end of the range is reached. Accordingly, the progress dialog box created by this component behaves the same way. If you want a different behavior, you must ensure that your current position does not exceed the upper limit.

The following functions are public functions which are specific to the progress dialog class:

BOOL Create(CWnd ****pWndParent* **= NULL)

Parameters

pWndParent Specifies the parent of the progress dialog. If NULL is used, then the progress dialog will default to using the application's main window (retrieved using the **AfxGetMainWnd** API function).

Return Value

TRUE if the dialog was created successfully. FALSE on failure.

Comments

Call this function to create the progress dialog. Although the progress dialog is created as a modeless dialog, it behaves as a modal dialog. This is accomplished by disabling the parent window, and enables you to continue processing. When the dialog box object is destroyed, the parent window is re-enabled.

BOOL CheckCancelButton()

Return Value

TRUE if the user has selected the **Cancel** button, otherwise false.

Comments

Call this function periodically to determine if the user has canceled the operation. The **Cancel** button is "reset" between invocations of this call, so the next call to `CheckCancelButton` will only return TRUE if the user has hit Cancel since the last time the `CheckCancelButton` function was called.

This function is available only if you selected the 'Cancel Button' option when inserting the progress dialog.

void SetStatus(LPCTSTR *lpszStatusText***)**

Parameters

lpszStatusText A string containing the status text to be displayed.

Call this function when you wish to update the status text.

This function is available only if you selected the **Status Text** option when inserting the progress dialog.

Using the Progress Dialog

You use the progress dialog box in the following manner:

1. Declare the progress dialog object.
2. Call `Create` to create the progress dialog.
3. Perform your time-consuming task. Periodically during your processing you can call a combination of the following functions: `StepIt`, `SetPos`, `CheckCancelButton`, `SetStatusText`.
4. Destroy the progress dialog object.

A simple example of this usage follows:

```
// Don't forget to include the header file for the progress dialog class:
#include "progdlg.h"

void OnCalculate()
{
  CProgressDlg dlg;

  dlg.Create();  // By default Create will disable the application's main
window
   // For a progress dialog with a range that contains 10 steps ( (upper-
lower) / step)
   // we would use a loop with ten calls to StepIt(). Note that the range
or the step size
  // could be changed programmatically with SetRange or SetStep.
  for(int i=0; i<10; i++)
  {
    if(dlg.CheckCancelButton())
      if(AfxMessageBox(_T("Are you sure you want to
Cancel?"),MB_YESNO)==IDYES)
        break;
    DoSingleCalculation(i);  // Some function or block of code that
performs a single
                                                // iteration of the processing
    dlg.StepIt();
   };
  // At this point either the operation has completed or the user has
cancelled.
}
```

Progress Dialog Component: Results

The Progress Dialog component makes the following changes to your project:

- A dialog resource is added to your project. The definition of this resource will depend on the options you select when inserting the component (for example, Status Text, Show Percentage, Cancel Button). You can use the Visual C++ dialog editor to customize this dialog resource.
- A header file is added to your project. This file contains the declaration of the progress dialog class. You will need to include this header file in any file where you intend to use the progress dialog class.
- An implementation file will be added to your project. This file contains the full implementation of the progress dialog class.

Progress Dialog Component: Class Page

The following options are available when you are inserting the progress dialog into your project:

Class Name Specify the name of the progress dialog's class.

Dialog Caption Use this control to specify the text you wish to set for the progress dialog's caption. You can change this text programmatically using the **CWnd::SetWindowText** function if desired.

Test Press this button at any time to see a preview of what the progress dialog will look like given the current settings.

Dialog Options:

- **Status Text** Select this option if you wish to have a static text control on your dialog for displaying status information for your user. Selecting this option will also add the `SetStatus` function to the progress dialog class to enable you to programmatically change this text.
- **Percentage** Select this option if you wish to have a percentage shown above the progress indicator on your dialog box.
- **Cancel Button** Select this option if you wish to have a **Cancel** button on your progress dialog. Selecting this option will also add the `CheckCancelButton` function to your class, enabling you to detect when the user has pressed the **Cancel** button.

Gauge Range

- **Lower** Specify the lower limit to be used in the progress indicator's range. This can be any value in the range 0 to 65,535.
- **Upper** Specify the upper limit to be used in the progress indicator's range. This can be any value in the range 0 to 65,535.
- **Step** Specify the Step value to be used by the progress dialog class's `StepIt` function. This can be any value in the range 1 to 65,535.

Change Press this button if you wish to change the implementation and header files which are to be used for the progress dialog class.

Progress Dialog Component: Change Files

Use the **Change Files** dialog box to change the name of the header and implementation files of the helper functions.

Header file Use this control to change the name of the header file.

Implementation file Use this control to change the name of the implementation file.

Browse Displays a dialog box for selecting a new drive and/or directory for the location of the header or implementation file.