Lab 6.3: Using Common Dialogs

Objectives

After completing this lab, you will be able to:

- Use a common font dialog box in an application.
- Use a common color-choice dialog box in an application.
- Modify color and font in a CRichEditView.

Prerequisites

Familiarity with the topics covered in this chapter.

Lab Setup

To run the solution to this lab, click this icon.

To see a demonstration of the solution to this lab, click this icon.

Estimated time to complete this lab: **15 minutes**.

Exercises

The following exercises provide practice working with the concepts and techniques covered in this chapter.

Exercise 1: Adding a Font Dialog

In this exercise, you will add a font dialog box to an application.

Exercise 2: Adding a Color Picker Dialog

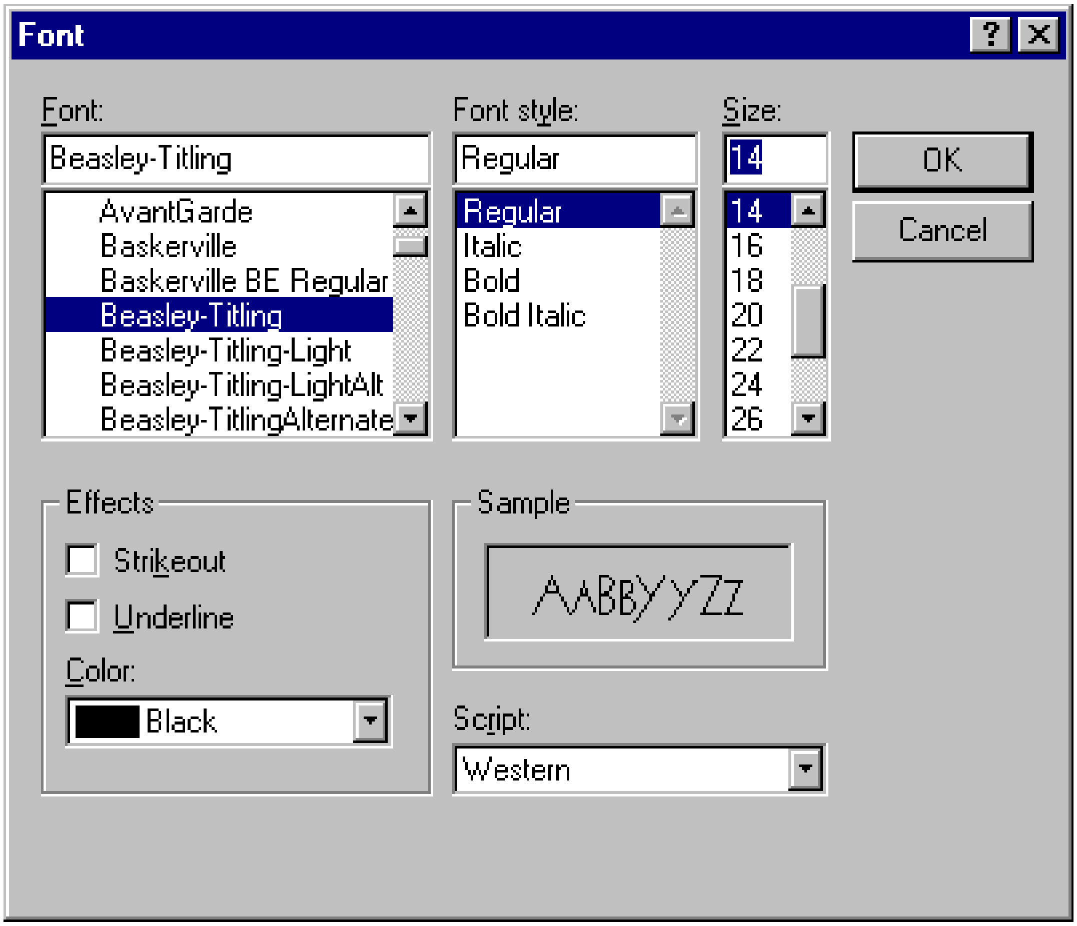
In this exercise, you will add a color-choice dialog box to an application.

For this lab, you will use the project that you created in Chapter 5, Lab 5: Adding a Shortcut Menu. You can copy this from \Labs\C06\Lab03\Baseline. The completed code for these exercises is in \Labs\C06\Lab03\Xxx, where Xxx is the exercise number.

Exercise 1: Adding a Font Dialog

If you do not have a starting point for this exercise, the code that forms the basis for this exercise is in \Labs\C06\Lab03\Baseline.

In this exercise, you will add a font dialog box to an application. When a user chooses the Font option from the Context menu, the application will display the common font dialog box, as shown in this figure.



Font: Beasley-Titling

Font style: Regular

Size: 14

☐ Strikeout

☐ Underline

Color: Black

Script: Western

Based on the user's choice, the font will be set in the pane of ShowDiff in which the user clicked.

▶ **Add a data member for the current font:**

1. Open DiffView.H.
2. In the protected implementation section, add a CHARFORMAT member to store the character-formatting characteristics.

```
CHARFORMAT   m_CharacterFormat;
```

3. Save DiffView.H.

▶ **Change the OnEditFont handler to show a font dialog box:**

1. When you display a font dialog box, you should initialize it to show the current font. Call CRichEditCtrl::GetDefaultCharFormat to get the format of the pane on which your user clicked.

```
GetRichEditCtrl().GetDefaultCharFormat(m_CharacterFormat);
```

2. The documentation for the **CFontDialog** constructor describes one of the two parameter sets. The Microsoft Foundation Class Library (MFC) 4 adds Rich Edit support with this parameter list.

```
CFontDialog::CFontDialog(const CHARFORMAT& charformat, DWORD dwFlags,
    CDC* pdcPrinter, CWnd* pParentWnd) : CCommonDialog(pParentWnd)
```

   Use this form to construct the dialog box. Show only screen fonts in the font list.

```
CFontDialog dlg(m_CharacterFormat,CF_SCREENFONTS);
```

3. Show the dialog box as a modal dialog box, and proceed if the user has chosen a font or style.

```
if (dlg.DoModal() == IDOK)
```

4. Get the chosen character formatting.

```
dlg.GetCharFormat(m_CharacterFormat);
```

5. Set the character formatting of the pane to the chosen formatting.

```
GetRichEditCtrl().SetDefaultCharFormat(m_CharacterFormat);
```

6. Save DiffView.Cpp.
7. Build and run ShowDiff to test your font dialog code.

The complete function follows:

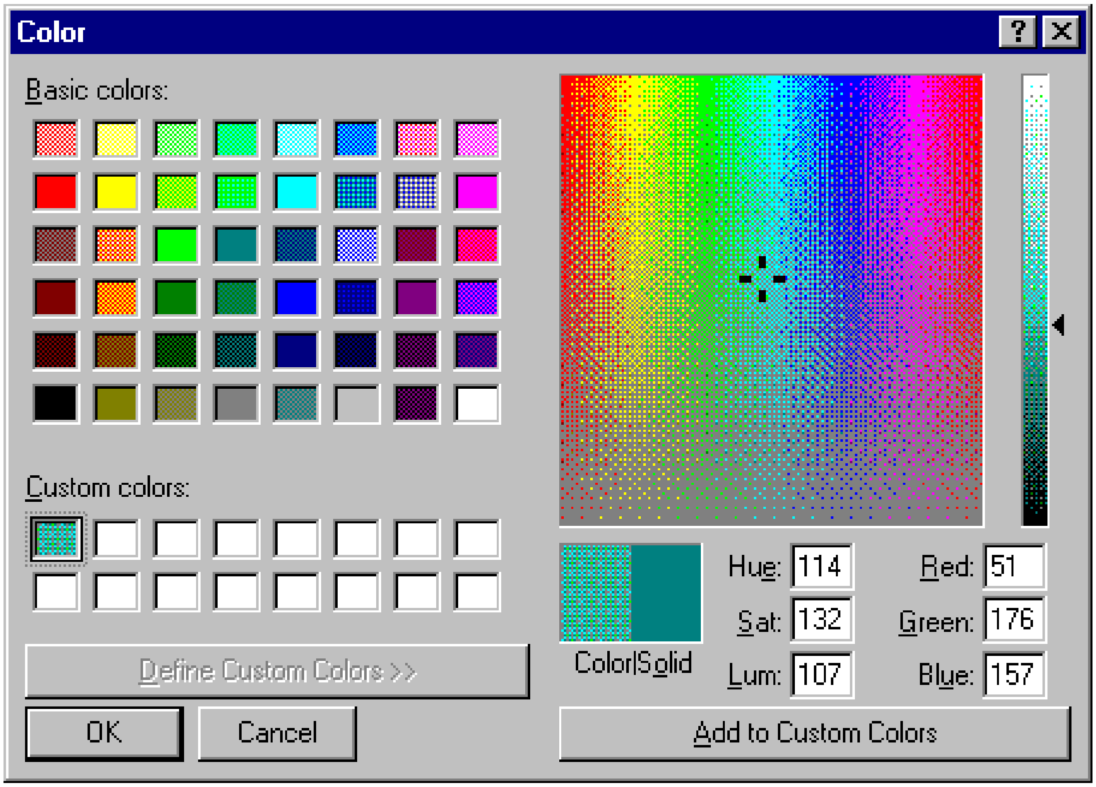
```
void CDiffView::OnEditFont()
{
    GetRichEditCtrl().GetDefaultCharFormat(m_CharacterFormat);

    CFontDialog dlg(m_CharacterFormat,CF_SCREENFONTS);

    if (dlg.DoModal() == IDOK)
    {
        dlg.GetCharFormat(m_CharacterFormat);
        GetRichEditCtrl().SetDefaultCharFormat(m_CharacterFormat);
    }

}
```

The completed code for this exercise is in \Labs\C06\Lab03\Ex01.

Exercise 2: Adding a Color Picker Dialog

If you do not have a starting point for this exercise, the code that forms the basis for this exercise is in \Labs\C06\Lab03\Ex01.

In this exercise, you will add a color-choice dialog box to an application. When a user chooses either of the Color options from the Context menu, the application will display the common color dialog box, which is shown in this figure.



Hue: 114
Sat: 132
Lum: 107
Red: 51
Green: 176
Blue: 157

Based on the user’s choice, the text color or background will be set in the pane of ShowDiff in which the user clicked.

▶ Add data members for display colors

1. Open DiffView.H.
2. In the protected implementation section, add COLORREF members to hold the colors for the background and the foreground:

```
COLORREF     m_ForegroundColor;
COLORREF     m_BackgroundColor;
```

3. Save DiffView.H.

▶ Initialize the new data members

1. Open DiffView.Cpp.
2. In the constructor, add the following code:

```
m_ForegroundColor = RGB (0, 0, 0);
m_BackgroundColor = RGB (255, 255, 255);
```

▶ Change the OnEditColorAdditions handler to show a color dialog box

1. When you display a color dialog box, you should initialize it to show the current color. Call **CRichEditCtrl::GetDefaultCharFormat** to get the format of the pane on which your user clicked.

```
GetRichEditCtrl().GetDefaultCharFormat(m_CharacterFormat);
```

2. The CHARFORMAT structure contains a COLORREF member, which holds the pen color for the format. Set this member to m_AdditionsColor.

```
m_CharacterFormat.crTextColor = m_AdditionsColor;
```

3. Use this member to construct a **CColorDialog** object.

```
CColorDialog dlg(m_CharacterFormat.crTextColor);
```

4. Show the dialog box as a modal dialog box, and proceed if the user has chosen a color.

```
if (dlg.DoModal() == IDOK)
```

5. Get the chosen color, set dwEffects to 0, and place a value in dwMask.

```
m_CharacterFormat.crTextColor = dlg.GetColor();
m_CharacterFormat.dwEffects = 0;
m_CharacterFormat.dwMask = CFM_COLOR;
```

   If the user creates a custom color, MFC will store that color in your application.

6. Store the chosen color in m_AdditionsColor.

```
m_AdditionsColor = m_CharacterFormat.crTextColor;
```

7. Set the character formatting of the pane with the changed color.

```
GetRichEditCtrl().SetDefaultCharFormat(m_CharacterFormat);
```

8. The complete **OnEditColorForeground** function is as shown in this sample code.

```
void CDiffView::OnEditColorForeground()
{
    GetRichEditCtrl().GetDefaultCharFormat(m_CharacterFormat);
    m_CharacterFormat.crTextColor = m_ForegroundColor;
    CColorDialog dlg(m_CharacterFormat.crTextColor);

    if (dlg.DoModal() == IDOK)
    {
        m_CharacterFormat.crTextColor = dlg.GetColor();
        m_CharacterFormat.dwEffects = 0;
        m_CharacterFormat.dwMask = CFM_COLOR;
        m_ForegroundColor = m_CharacterFormat.crTextColor;
        GetRichEditCtrl().SetDefaultCharFormat(m_CharacterFormat);
    }
}
```

9. Code the **OnEditColorBackground** function to use the **CRichEditView::SetBackgroundColor** data member as follows.

```
void CDiffView::OnEditColorBackground()
{
    CColorDialog dlg(m_BackgroundColor);

    if (IDOK == dlg.DoModal())
    {
        m_BackgroundColor = dlg.GetColor();
        GetRichEditCtrl().SetBackgroundColor(FALSE, m_BackgroundColor);
    }
}
```

10. Save DiffView.Cpp.
11. Build and run ShowDiff

The completed code for this exercise is in \Lab\C06\Lab03\Ex02.