BrowseDr unit

This unit provides a component that displays a standard Windows 95/NT 4 dialog containing the user's system in a heirarchial manner and allows a selection to be made. It is a wrapper for the SHBrowseForFolder() API, which is quite messy to use directly. Also provided is an editor which allows you to display the dialog at design time with the selected options.

Note:
This component requires Delphi 3 or Delphi v2.01's ShlObj unit. If you have Delphi 2.00, you can get the equivalent using Pat Ritchey's ShellObj unit. It is freely available on his web site at http://ourworld.compuserve.com/homepages/PRitchey/. Both Borland's ShlObj unit and Pat's ShellObj unit contain errors that should be fixed. I have included instructions on how to do this. They are in the included ShellFix.txt file. Delphi 3's ShlObj unit does not have any errors that I am currently aware of.

Components
TBrowseDialogEditor
TBrowseDirectoryDlg

Types
TBDSelChangedEvent
TBrowseFlag
TBrowseFlags
TRootID

Constants
BIF_BROWSEINCLUDEFILES
CSIDL_DESKTOPEXPANDED
CSIDL_INTERNET

TBrowseDialogEditor Component

Unit
BrowseDr

Description
TBrowseDialogEditor is a component editor provided so that you can see the effects of various settings have on a TBrowseDirectoryDlg component at design-time. To use it, simply right-click on a TBrowseDirectoryDlg component and select "Test Dialog". No further documentation is provided for this editor, but the complete source code is available in the BrowseDr.pas unit.

TBrowseDirectoryDlg Component

Properties Methods Events

Unit
BrowseDr

Description
TBrowseDirectoryDlg provides a component that displays a standard Windows 95/NT 4 dialog containing the user's system in a heirarchial manner and allows a selection to be made. It is a wrapper for the SHBrowseForFolder() API, which is quite messy to use directly.

Properties

Run-time only
Key properties

Caption	ImageIndex	SelectionPIDL
Center	Options	ShowSelectionInStatus
DisplayName	Parent	StatusText
EnableOKButton	Root	Title
FitStatusText	Selection	

Methods

Key methods
Execute

Events

Key events

OnCreate	OnSelChanged

Caption property

Applies to
TBrowseDirectoryDlg

Declaration
```
property Caption: string;
```

Description
Specifies the text in the dialog's caption bar. Use Caption to specify the text that appears in the browse folder dialog's title bar. If no value is assigned to Title, the dialog has a title based on the Options property.

For example, if **[bfPrinters]** was set as the Options value, the title would be "Browse for Printer".

Center property

Applies to
TBrowseDirectoryDlg

Declaration
```
property Center: boolean;
```

Description
Indicates whether the dialog should be centered on the screen or shown in a default, system determined location.

DisplayName property

Applies to
TBrowseDirectoryDlg

Declaration
```
property DisplayName: string;
```

Description
DisplayName is run-time, read-only property that returns the display name of the selection. It only has meaning after the dialog has been executed and the user has made a selection. It returns the "human readable" form of the selection. This generally is the same as the Selection property when it is a file path, but in the case of items such as the Control Panel which do not have a path, Selection is blank. In this case, the only way to access the users' selection is to use the SelectionPIDL property. That doesn't provide an easy way of presenting a textual representation of what they chose, but this property will do that for you.

If, for example, the user chose the Control Panel folder, the Selection property would be blank, but DisplayName would be "Control Panel". You could not actually use this value to get to the Control Panel, for that you need to use the SelectionPIDL property and various Shell Namespace API functions.

EnableOKButton property

Example

Applies to

TBrowseDirectoryDlg

Declaration

```
property EnableOKButton: boolean;
```

Description

This property enables or disables the OK button on the browse folders dialog. This allows control over whether a selection can be made or not. You can modify this value while the dialog is displayed from the OnSelChanged event. This allows you to control whether the user can select an item based on what the current selection is.

FitStatusText property

Applies to
TBrowseDirectoryDlg

Declaration
```
property FitStatusText: boolean;
```

Description
Indicates whether the StatusText string should be shortened to make it fit in available status text area. The status text area is only large enough to hold one line of text, and if the text is too long for the available space, it will simply be chopped off. However, if this property is set to TRUE, the text will be shortened using an ellipsis (`"..."`).

For example, if the status text property were `"C:\Windows\Start Menu\Programs\Applications\Microsoft Reference"`, it could be shortened to `"C:\...\Start Menu\Programs\Applications\Microsoft Reference"` depending on the screen resolution and dialog font size.

ImageIndex property

Applies to
TBrowseDirectoryDlg

Declaration
```
property ImageIndex: integer;
```

Description
After a selection has been made in the dialog, this property will contain the index into the system image list of the selected node. See the demo application for an example how this can be used.

Options property

Applies to
TBrowseDirectoryDlg

Declaration
```
property Options: TBrowseFlags;
```

Description
Options is a set of TBrowseFlag items that controls what is allowed to be selected and expanded in the tree. It can be a combination of any (or none) of the following:

bfDirectoriesOnly: Only returns file system directories. If the user selects folders that are not part of the file system, the OK button is grayed.
bfDomainOnly: Does not include network folders below the domain level in the dialog.
bfAncestors: Only returns file system ancestors (items which contain files, like drives). If the user selects anything other than a file system ancestor, the OK button is grayed.
bfComputers: Shows other computers. If anything other than a computer is selected, the OK button is disabled.
bfPrinters: Shows all printers. If anything other than a printers is selected, the OK button is disabled.
bfIncludeFiles: Show non folder items that exist in the folders.

Parent property

Applies to
TBrowseDirectoryDlg

Declaration
```
property Parent: TWinControl;
```

Description
The window component that is the browse dialog's parent window. By assigning a value to this property, you can control the parent window independant of the form that the component exists on.

You do not normally need to assign any value to this property as it will use the form that contains the component by default.

Root property

Applies to
TBrowseDirectoryDlg

Declaration
```
property Root: TRootID;
```

Description
Specifies the item that is to be treated as the root of the tree display.

idDesktop: Windows desktop -- virtual folder at the root of the name space.
idInternet: Internet Explorer -- virtual folder of the Internet Explorer.
idPrograms: File system directory that contains the user's program groups (which are also file system directories).
idControlPanel: Control Panel -- virtual folder containing icons for the control panel applications.
idPrinters: Printers folder -- virtual folder containing installed printers.
idPersonal: File system directory that serves as a common respository for documents.
idFavorites: Favorites folder -- virtual folder containing the user's Internet Explorer bookmark items and subfolders.
idStartup: File system directory that corresponds to the user's Startup program group.
idRecent: File system directory that contains the user's most recently used documents.
idSendTo: File system directory that contains Send To menu items.
idRecycleBin: Recycle Bin -- file system directory containing file objects in the user's recycle bin. The location of this directory is not in the registry; it is marked with the hidden and system attributes to prevent the user from moving or deleting it.
idStartMenu: File system directory containing Start menu items.
idDesktopDirectory: File system directory used to physically store file objects on the desktop (not to be confused with the desktop folder itself).
idDrives: My Computer -- virtual folder containing everything on the local computer: storage devices, printers, and Control Panel. The folder may also contain mapped network drives.
idNetwork: Network Neighborhood -- virtual folder representing the top level of the network hierarchy.
idNetHood: File system directory containing objects that appear in the network neighborhood.
idFonts: Virtual folder containing fonts.
idTemplates: File system directory that serves as a common repository for document templates.
idCommonStartMenu: File system directory that contains the programs and folders that appear on the Start menu for all users (NT only).
idCommonPrograms: File system directory that contains the directories for the common program groups that appear on the Start menu for all users (NT only).
idCommonStartup: File system directory that contains the programs that appear in the Startup folder for all users. The system starts these programs whenever any user logs on to Windows NT (NT only).
idCommonDesktopDirectory: File system directory that contains files and folders that appear on the desktop for all users (NT only).
idAppData: File system directory that contains data common to all applications.
idPrintHood: File system directory containing object that appear in the printers folder.
idDesktopExpanded: Same as idDesktop except that the root item is already expanded when the dialog is initally displayed.

NOTE: idCommonStartMenu, idCommonPrograms, idCommonStartup, and idCommonDesktopDirectory only have effect when the dialog is being displayed on an NT system. On Windows 95, these values will be mapped to thier "non-common" equivalents, i.e. idCommonPrograms will become idPrograms.

Selection property

Applies to
TBrowseDirectoryDlg

Declaration
```
property Selection: string;
```

Description
The selected item in the browse folder dialog.

Setting this before calling the Execute method will cause the assigned value to be initially selected when the dialog is first displayed if the item exists. If it does not exist, the root item will be selected. If this value is blank, the SelectionPIDL item will be used instead to set the initally selected item.

After the Execute method returns, you can read this value to determine what item the user selected, unless that item does not have a string representation (Control Panel, for example). In this case, you have to use the SelectionPIDL for the item the user selected, but you can use the DisplayName property to retrieve a string that can be shown to the user to identify the selected item.

SelectionPIDL property

Example

Applies to
TBrowseDirectoryDlg

Declaration
```
property SelectionPIDL: PItemIDList
```

Description
An alternative to the Selection property. Use this property if the item you are interested in does not have a path (Control Panels, for example). The most common way to retrieve a value for this property is to use the SHGetSpecialFolderLocation Windows API function. Once you have assigned a value to this property, it is "owned" by the component. That is, the component will take care of freeing it when it is no longer needed.

When setting this property before calling the Execute method, it will only be used if the Selection property is blank. If Selection is not blank, it will be used instead.

Upon return from the Execute method, this property will contain the PItemIDList of the item the user selected. In some cases, this will the only way to get the user's choice since items such as Control Panel do not have a string that can be placed in the Selection property.

ShowSelectionInStatus property

Applies to
TBrowseDirectoryDlg

Declaration
```
property ShowSelectionInStatus: boolean;
```

Description
Automatically shows the current selection in the status text area of the dialog.

StatusText property

Example

Applies to

TBrowseDirectoryDlg

Declaration

```
property StatusText: string;
```

Description

A string that is displayed directly above the tree view control and just under the Title text in the dialog box. This string can be used for any purpose such as to specify instructions to the user, or show the full path of the currently selected item. You can modify this value while the dialog is displayed from the the OnSelChanged event.

If StatusText is blank when the Execute method is called, the dialog will not have a status text area and assigning to the StatusText property will have no effect.

Title property

Applies to
TBrowseDirectoryDlg

Declaration
```
property Title: string;
```

Description
Specifies the text to appear at the top of the dialog above the tree control. There is enough room for two lines of text, and it will be word-wrapped for you automatically.

Generally, this is used to provide user instructions or as a title for the StatusText property.

Execute method

Applies To
TBrowseDirectoryDlg

Declaration
```
function Execute: boolean;
```

Description
Displays the browser folders dialog. It returns TRUE if user selected an item and pressed OK, otherwise it returns FALSE.

OnCreate event

Applies To
TBrowseDirectoryDlg

Declaration
```
property OnCreate: TNotifyEvent;
```

Description
The OnCreate event is fired when dialog has been created, but just before it is displayed to the user.

OnSelChanged event

Example

Applies To

TBrowseDirectoryDlg

Declaration

```
TBDSelChangedEvent = procedure(Sender: TObject; const NewSel: string;
NewSelPIDL: PItemIDList) of object;
property OnSelChanged: TBDSelChangedEvent;
```

Description

The OnSelChange event is fired every time a new item is selected in the tree.

The Sender parameter is the TBrowseDirectoryDlg object whose event handler is called. The NewSel parameter is the text representation of the new selection. The NewSelPIDL is the new PItemIDList representation of the new selection.

NOTE: You will need to add ShlObj to your uses clause if you define a handler for this event.

TBDSelChangedEvent type

Unit
BrowseDr

Declaration
```
TBDSelChangedEvent = procedure(Sender: TObject; const NewSel: string;
NewSelPIDL: PItemIDList) of object;
```

Description
TBDSelChangedEvent is used for events associated with TBrowseDirectoryDlg's OnSelChanged event.

The Sender parameter is the TBrowseDirectoryDlg object whose event handler is called. The NewSel parameter is the text representation of the new selection. The NewSelPIDL is the new PItemIDList representation of the new selection.

BIF_BROWSEINCLUDEFILES constant

Unit
BrowseDr

Declaration
```
BIF_BROWSEINCLUDEFILES = $4000;
```

Description
This constant was missing from the Delphi 2 units, but was added to Delphi 3. It causes files to be included in the tree as well as folders.

CSIDL_DESKTOPEXPANDED constant

Unit
BrowseDr

Declaration
```
CSIDL_DESKTOPEXPANDED = $FEFE;
```

Description
This folder identifer is undocumented, but should work for a long time since the highest ID is currently around 30 or so. It is used to open the tree already expanded with the desktop as the root item.

CSIDL_INTERNET constant

Unit
BrowseDr

Declaration
```
CSIDL_INTERNET = $0001;
```

Description
This is a newly documented folder identifier that is not in the Delphi units yet. You can use it with any of the Win32 Shell API functions that wants a CSIDL_* identifier such as SHGetSpecialFolderLocation.

Example

Attaching this code to the OnSelChange event will cause the OK button to be disabled when the current selection is less than 11 characters. A mean thing to do, but it's only an example.

```
procedure TForm1.BrowseDirectoryDlgSelChanged(Sender: TObject; const NewSel:
string);
begin
  // NewSel has the full selection.  Only allow items greater than 10
characters to be selected.
  BrowseDirectoryDlg.EnableOKButton := Length(NewSel > 10);
end;
```

Example

Attaching this code to the OnSelChange event will cause the dialog to display the full selection in a status text area of the dialog.

```
// Set Title property to "The current selection is:" in the Object Insector
// Set StatusText property to "Dummy Value" in the Object Insector
procedure TForm1.BrowseDirectoryDlgSelChanged(Sender: TObject; const NewSel:
string);
begin
  // NewSel has the full selection
  BrowseDirectoryDlg.StatusText := NewSel;
end;
```

Example

This example illustrates how you could set the initial selection to the Start Menu folder when the root item is the desktop.

```
// ShlObj must be in your uses clause.
var
  StartPIDL: PItemIDList;
begin
  // Get the PIDL to the special location
  SHGetSpecialFolderLocation(Handle, CSIDL_STARTMENU, StartPIDL);
  BrowseDirectoryDlg.Root := idDesktop;
  // SelectionPIDL only works if Selection is blank!
  BrowseDirectoryDlg.Selection := '';
  // Give it the PIDL.
  BrowseDirectoryDlg.SelectionPIDL := TempPIDL;

  BrowseDirectoryDlg.Execute;

  // Normally, you would have to free the PIDL when finished, but the
  // TBrowseDirectoryDlg component takes care of this.
end;
```

Example

Attaching this code to the OnSelChange event will cause the dialog to display the full selection in a status text area of the dialog.

```
// Set Title property to "The current selection is:" in the Object Insector
// Set StatusText property to "Dummy Value" in the Object Insector
procedure TForm1.BrowseDirectoryDlgSelChanged(Sender: TObject; const NewSel:
string);
begin
  // NewSel has the full selection
  BrowseDirectoryDlg.StatusText := NewSel;
end;
```

TBrowseFlag type

Unit
BrowseDr

Declaration
```
TBrowseFlag = (
    bfDirectoriesOnly, bfDomainOnly, bfAncestors, bfComputers, bfPrinters,
    bfIncludeFiles
   );
```

Description
These are equivalent to the BIF_* constants in the Win32 API. They are used to specify what items can be expanded, and what items can be selected by combining them in a set in the Options property.

bfDirectoriesOnly: Only returns file system directories. If the user selects folders that are not part of the file system, the OK button is grayed.
bfDomainOnly: Does not include network folders below the domain level in the dialog.
bfAncestors: Only returns file system ancestors (items which contain files, like drives). If the user selects anything other than a file system ancestor, the OK button is grayed.
bfComputers: Shows other computers. If anything other than a computer is selected, the OK button is disabled.
bfPrinters: Shows all printers. If anything other than a printers is selected, the OK button is disabled.
bfIncludeFiles: Show non folder items that exist in the folders.

TBrowseFlags type

Unit
BrowseDr

Declaration
```
TBrowseFlags = set of TBrowseFlag;
```

Description
A set of TBrowseFlag items.

TRootID type

Unit
BrowseDr

Declaration
```
TRootID = (
    idDesktop, idInternet, idPrograms, idControlPanel, idPrinters, idPersonal,
    idFavorites, idStartup, idRecent, idSendTo, idRecycleBin, idStartMenu,
    idDesktopDirectory, idDrives, idNetwork, idNetHood, idFonts, idTemplates,
    idDesktopExpanded
   );
```

Description
This enumerated type is the equivalent of the CSIDL_* constants in the Win32 API. They are used to specify the root of the heirarchy tree.

idDesktop: Windows desktop -- virtual folder at the root of the name space.
idInternet: Internet Explorer -- virtual folder of the Internet Explorer.
idPrograms: File system directory that contains the user's program groups (which are also file system directories).
idControlPanel: Control Panel -- virtual folder containing icons for the control panel applications.
idPrinters: Printers folder -- virtual folder containing installed printers.
idPersonal: File system directory that serves as a common respository for documents.
idFavorites: Favorites folder -- virtual folder containing the user's Internet Explorer bookmark items and subfolders.
idStartup: File system directory that corresponds to the user's Startup program group.
idRecent: File system directory that contains the user's most recently used documents.
idSendTo: File system directory that contains Send To menu items.
idRecycleBin: Recycle Bin -- file system directory containing file objects in the user's recycle bin. The location of this directory is not in the registry; it is marked with the hidden and system attributes to prevent the user from moving or deleting it.
idStartMenu: File system directory containing Start menu items.
idDesktopDirectory: File system directory used to physically store file objects on the desktop (not to be confused with the desktop folder itself).
idDrives: My Computer -- virtual folder containing everything on the local computer: storage devices, printers, and Control Panel. The folder may also contain mapped network drives.
idNetwork: Network Neighborhood -- virtual folder representing the top level of the network hierarchy.
idNetHood: File system directory containing objects that appear in the network neighborhood.
idFonts: Virtual folder containing fonts.
idTemplates: File system directory that serves as a common repository for document templates.
idCommonStartMenu: File system directory that contains the programs and folders that appear on the Start menu for all users (NT only).
idCommonPrograms: File system directory that contains the directories for the common program groups that appear on the Start menu for all users (NT only).
idCommonStartup: File system directory that contains the programs that appear in the Startup folder for all users. The system starts these programs whenever any user logs on to Windows NT (NT only).
idCommonDesktopDirectory: File system directory that contains files and folders that appear on the

desktop for all users (NT only).

idAppData: File system directory that contains data common to all applications.

idPrintHood: File system directory containing object that appear in the printers folder.

idDesktopExpanded: Same as idDesktop except that the root item is already expanded when the dialog is initally displayed.

NOTE: idCommonStartMenu, idCommonPrograms, idCommonStartup, and idCommonDesktopDirectory only have effect when the dialog is being displayed on an NT system. On Windows 95, these values will be mapped to thier "non-common" equivalents, i.e. idCommonPrograms will become idPrograms.