Help items

The Component UI API supports navigation help content and F1 context help via a NavigationItemUIInterface.

Navigation based help allows you to associate help with a particular navigation hierarchy in the navigation tree. If you have registered a help item for a particular navigation item then when a user navigates to that location in the tree the framework will query you for help content for that item. F1 help allows you to supply help in your dialogs and panels that is displayed when the user presses F1.

Navigation Help

Navigation help can be either static or dynamic. Static help is set when you insert the item with NavInsertItem but dynamic help is queried for by the framework when the user navigates to an associated navigation tree item.

To get dynamic help appearing there are two things you need to do.

First, register a help item with a Navigation Path that contains the path of the associated tree item. For instance, if the tree item was registered with the path ***NAVIGATIONTREE\ControlPanel\MyComponent*** then the path you need to use for the main content of the help item is ***HELP\NAVIGATIONTREE\ControlPanel\MyComponent***

Secondly, register child help items for each section of the help pane that you wish to place help content into.

e.g.

HELP\NAVIGATIONTREE\ControlPanel\MyComponent\Main

HELP\NAVIGATIONTREE\ControlPanel\MyComponent\Related

HELP\NAVIGATIONTREE\ControlPanel\MyComponent\Header

The different sections allow you to place help content in different parts of the help pane. The sections available are:

Type	Description
Header	Places content at the top of the help pane. Used for small content, such as links. The WMC puts a link to the WinGate Help there.
Main	The section where the main content of your help will reside
Related	Places content under the "Related Topics" section

Specifying the help item path

In order to ensure that the path you use for your help item matches the tree item, you can get the navigation path for the tree item with NavGetItem and use the strFQN field to build the path passed to NavInsertItem for the help item. The following example inserts an item under the Control Panel item in the tree and associates dynamic help with it.

> The NavItemHelpers code in sdk\helpers has more convenient ways to manage navigation items.

```
static NavigationItemUIInterface notificationInterface;

void SetupNotificationInterface()
{
    // Set up our notification interface so we can receive messages relating to the
the navigation item we will insert into the tree
    memset(&notificationInterface, 0, sizeof(notificationInterface));
    notificationInterface.dwSize                    =
sizeof(notificationInterface);
    notificationInterface.NavItemCreateWindow       = NavItemCreateWindow;
// Called when the framework wants a window from us
    notificationInterface.NavItemDrawItem           = NULL;
    notificationInterface.NavItemNotify             = NavItemNotify;
// For receiving general messages
    notificationInterface.NavItemGetHelpContent     = NavItemGetHelpContent;
// For dynamic help content in the help pane of the UI
    notificationInterface.NavItemFreeHelpContent    = NavItemFreeHelpContent;
```

```
    notificationInterface.NavItemShowContextHelp    = NavItemShowContextHelp;
// For F1 help
    notificationInterface.NavItemKeyboardNavigate   = NavItemKeyboardNavigate;
// To handle custom keyboard navigation within our panel. (optional)
}

// Called below
NavHandle CreateTreeItem()
{
    // Get the correct location to insert our new item under. In this case it will
be the "Control Panel" node. This has a well known alias that we can use.
    NAVSETITEM controlPanelItem;
    if ( !NavGetItem(_T("{ControlPanel}"), &controlPanelItem) )
    {
        return NULL;                                        // Couldn't find it, so
we won't go any further
    }

    NAVSETITEM treeItem;
    memset(&treeItem, 0, sizeof(treeItem));
    treeItem.size = sizeof(treeItem);
    treeItem.flags = NI_FLAG_CUSTOMCONTAINER;               // This is a custom
container where we control the panel.
    treeItem.lParam = NULL;                                 // Set any special
context data that you want to be sent with messages to the notificationInterface
    treeItem.strLabel = L"Ad Blocker";
    treeItem.strDescription = L"This component adds an HTTP Filter that blocks
advertising";
    treeItem.hModule = AfxGetResourceHandle();              // Where to get
resources from
    treeItem.nIconId = iconResourceId;                      // Some resource id of
the icon to display in the navigation tree
    treeItem.EventMask = 0;                                 // No special events.
NI_EVENT_ACTIVATE, NI_EVENT_DEACTIVATE are always sent. See NavInsertItem for more
information
    treeItem.pInterface = notificationInterface;            // The interface we are
wanting to use to handle events from this item
    treeItem.strAlias = L"{MyCompanyAdBlocker}";          // Adds an alias. Makes
it easier to find and compare navigation items later.
    if ( !NavInsertItem(controlPanelItem.hItem, L"MyCompanyAdBlockerPath",
&treeItem) )   // The path "MyCompanyAdBlockerPath" is an internal name the user
never sees but must be unique under a given parent.
    {
        return NULL;
    }
    return treeItem.hItem;         // NavInsertItem sets hItem if successful
}

std::wstring GetFQNOfTreeItem(NavHandle treeItem)
{
 NAVSETITEM itemData;
 NavGetItemEx(treeItem, &itemData);
 return itemData.strFQN;
}

NavHandle GetHelpRoot()
{
 NAVSETITEM item;
 if(NavGetItem(L"HELP", &item))
 {
  return item.hItem;
 }
 return NULL;
```

```
}

// Error checking removed for brevity
void SetUpNavigationItems()
{
 SetupNotificationInterface();
 NavHandle treeItem = CreateTreeItem();
 std::wstring treeItemFqn = GetFQNOfTreeItem(treeItem);

 NavHandle helpRoot = GetHelpRoot();

 NAVSETITEM helpItem;
 memset(&helpItem, 0, sizeof(helpItem));
 helpItem.size = sizeof(helpItem);
 helpItem.strLabel = L"My help label";
 helpItem.strDescription = L"Help for my component";
 helpItem.flags = NI_FLAG_DYNAMICHELP;      // We want to be called dynamically.
 helpItem.EventMask = NI_EVENT_NOTIFY;       // Need to set this so our callback
for dynamic help is processed
 helpItem.pInterface = &notificationInterface;    // Interface to handle callbacks
to ask for dynamic help
 NavInsertItem(helpRoot, (treeItemFqn + L"\\Main").c_str(), helpItem);  // Insert
the item as a child of HELP, with the same path as the tree item for the Main
```

```
section
}
```