



Citrix NFuse Elite Menu CDA Architecture

By Citrix Consulting Services

Citrix Systems, Inc.



Notice

The information in this publication is subject to change without notice.

THIS PUBLICATION IS PROVIDED "AS IS" WITHOUT WARRANTIES OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING ANY WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT. CITRIX SYSTEMS, INC. ("CITRIX"), SHALL NOT BE LIABLE FOR TECHNICAL OR EDITORIAL ERRORS OR OMISSIONS CONTAINED HEREIN, NOR FOR DIRECT, INCIDENTAL, CONSEQUENTIAL OR ANY OTHER DAMAGES RESULTING FROM THE FURNISHING, PERFORMANCE, OR USE OF THIS PUBLICATION, EVEN IF CITRIX HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES IN ADVANCE.

This publication contains information protected by copyright. Except for internal distribution, no part of this publication may be photocopied or reproduced in any form without prior written consent from Citrix.

The exclusive warranty for Citrix products, if any, is stated in the product documentation accompanying such products. Citrix does not warrant products other than its own.

Product names mentioned herein may be trademarks and/or registered trademarks of their respective companies.

Copyright © 2002 Citrix Systems, Inc., 851 West Cypress Creek Road, Ft. Lauderdale, Florida 33309-2009 U.S.A. All rights reserved.

Version History		
Version 1.2	Ed York	June 2002

Table of Contents

TABLE OF CONTENTS	III
OVERVIEW.....	1
INTRODUCTION.....	2
AUDIENCE	2
PREREQUISITES	2
OVERVIEW OF MENU CDA ARCHITECTURE	3
MENU CDA COMPONENTS	4
MENU CDA PROCESS FLOW	4
DETAILED MENU CDA ARCHITECTURE.....	6
MENU CDA IS EXECUTED.....	6
GENERATEMENU() METHOD IS EXECUTED	9
<i>CDA Caching is Disabled</i>	9
<i>CDA Header is Hidden</i>	9
<i>Menu content is retrieved and placed in XML string</i>	9
<i>(Optional) Menu content is cached in the Session</i>	10
<i>Conversion.xsl is applied to XML string</i>	11
<i>Menu.xsl is applied to Formatted XML string</i>	11
<i>HTML is written to browser</i>	12
CUSTOMIZING THE MENU CDA.....	13
ADDING SHOW/HIDE FUNCTIONALITY TO THE TREE MENU	14
DEBUGGING THE MENU CDA.....	15
GENERATING A DEBUG TEXT FILE.....	15
APPENDIX.....	17
SUMMARY OF MENU CDA COMPONENTS.....	17
<i>sbMenu CDA</i>	17
<i>COM DLL (CTXCDANavigation.dll)</i>	17
<i>Conversion.xsl file</i>	17
<i>Menu.xsl file</i>	18



<i>Tree.xsl</i>	18
<i>Menu-main.xsl</i>	18
<i>Menu-spacers.xsl</i>	19
<i>Menu-dropdowns.xsl</i>	19
<i>Menu-scripts.xsl</i>	19
XML RETRIEVED FROM CDS OBJECTS.....	19
XML OUTPUTTED BY TRANSFORMATION.XSL.....	21

Overview

Citrix® NFuse™ Elite is the simple, powerful access portal server that provides secure aggregation of applications and information. By leveraging the power of Citrix NFuse Classic to enable Web-based access to Citrix MetaFrame applications, Citrix NFuse Elite expands beyond client-server systems to include access to Web-based applications, legacy applications, internal and external content, and other services, such as syndicated content feeds.

Citrix NFuse Elite provides its content through scripted applications called Content Delivery Agents (CDAs). CDAs provide the mechanism by which information is brought into the portal. Citrix NFuse Elite ships with a series of standard CDAs such as Login, Header, Footer, and Menu. These CDAs are completely customizable, and additional CDAs can be developed and integrated within the portal.

The Menu CDA is used to create and display the navigation menu for the portal. Citrix NFuse Elite provides two types of navigation menus to be used within a portal: drop-down and tree. The drop-down menu provides portal navigation at the top of each page using a series of drop-down menus. The tree menu provides portal navigation at the left of each page using a tree-like interface. The portal administrator specifies the type of navigation menu used by the portal when the portal is created within the Portal Management Console (PMC).

The Menu CDA generates the content for the navigation menu by retrieving portal settings from the standard objects available within a CDA. Using the standard CDA objects, the Menu CDA retrieves the content for the menu and creates an XML file to store the content. This XML file is then transformed to HTML by using a series of XSL transformations. Since this code is based on Internet industry standards, the Menu CDA is customizable and extendible to fit any business needs.

Introduction

The purpose of this document is to present the architecture of the Menu CDA within Citrix NFuse Elite. This document outlines the components of the Menu CDA, explains how the menu content is generated and displayed, and gives tips on how to further customize the Menu CDA.

Audience

This document is directed towards developers familiar with Citrix NFuse Elite and that have CDA development, XML, and XSL experience. Developers can use this document to get an overview of the Menu CDA components and gain an understanding of how to update the components to customize the presentation of the navigation menu.

Prerequisites

In order to work with the Menu CDA, a developer needs to possess skills in the following technical areas:

- XML
- XSLT
- HTML
- JavaScript
- CDA Development (including CDA server-side scripting)

Overview of Menu CDA Architecture

The Menu CDA generates and displays the navigation menu for the NFuse Elite portal. When the Menu CDA is executed, the CDA calls a COM object to generate the content for the menu. The COM object retrieves the content for the menu, generates the HTML for the menu, and writes the HTML to the browser. The diagram below illustrates this process.

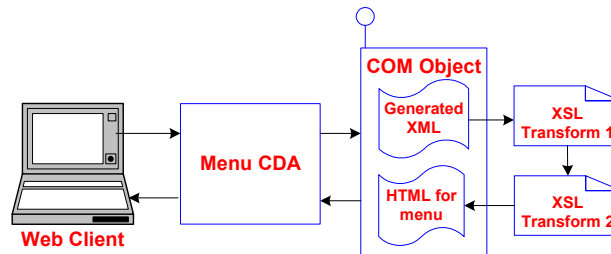


Figure 1 - Menu CDA Architecture

The COM object retrieves the content for the menu using a series of NFuse Elite CDS objects. When the CDA calls the COM object, the CDA passes in a series of CDS objects to the COM object such as Application, Session, Request, and Response. This allows the COM object to access the properties and methods of the CDS objects directly, and to use them to retrieve the content for the menu.

When the COM object retrieves the content for the menu, it creates an XML string containing the menu content. This XML string contains the list of folders, pages, CDAs, themes, and ICA applications that will be accessible from the menu.

The COM object then applies two XSL transformations to the XML string to create the HTML for the menu. The XSL transformations used by the COM object are contained within external XSL files and are located on the Agent Server. When the CDA calls the COM object, the CDA passes in a parameter specifying the name and location of these XSL files.

The first XSL transformation applied to the XML string is called Conversion.xml. This XSL file converts the XML string into a new “formatted” XML string. Conversion.xml organizes the XML into how the navigation menu will be presented, such as by grouping elements together and placing elements within a group in alphabetical order.

The second XSL transformation that is applied is called Menu.xml. This XSL file directs how the XML outputted by the Conversion.xml transformation is converted into HTML. The Menu.xml file imports a series of additional XSL files, such as Tree.xml, Menu-main.xml, and Menu-dropdowns.xml. Using a parameter that is passed in, the Menu.xml file determines the type of navigation menu to create. If a tree menu is specified, the Menu.xml file applies the Tree.xml transformation to create the HTML for a tree menu. If a drop-down menu is specified, the Menu.xml file applies the Menu-main.xml and Menu-dropdowns.xml transformations to create the HTML for the drop-down menu. After the HTML is generated, it is then written to the browser.

Menu CDA Components

The Menu CDA architecture consists of the following components:

- **Menu CDA code** – calls the COM DLL to generate and display the navigation menu
- **COM DLL (CTXCDANavigation.dll)** – retrieves the menu content using passed in CDS objects and places this content in an XML string. It then applies two XSL transformations (Conversion.xsl and Menu.xsl) to this XML string to generate the HTML for the menu.
- **Conversion.xsl file** – the first transformation applied to the XML string generated by the COM object. It converts the inputted XML string into a new “formatted” XML string. The formatted XML string is organized into how the navigation menu will be presented, such as by grouping elements together and listing elements within a group in alphabetical order.
- **Menu.xsl file** – the transformation applied to the XML outputted by the Conversion.xsl file. This transformation directs how the inputted XML is converted into HTML. Menu.xsl imports additional XSL files to actually perform the HTML conversion.

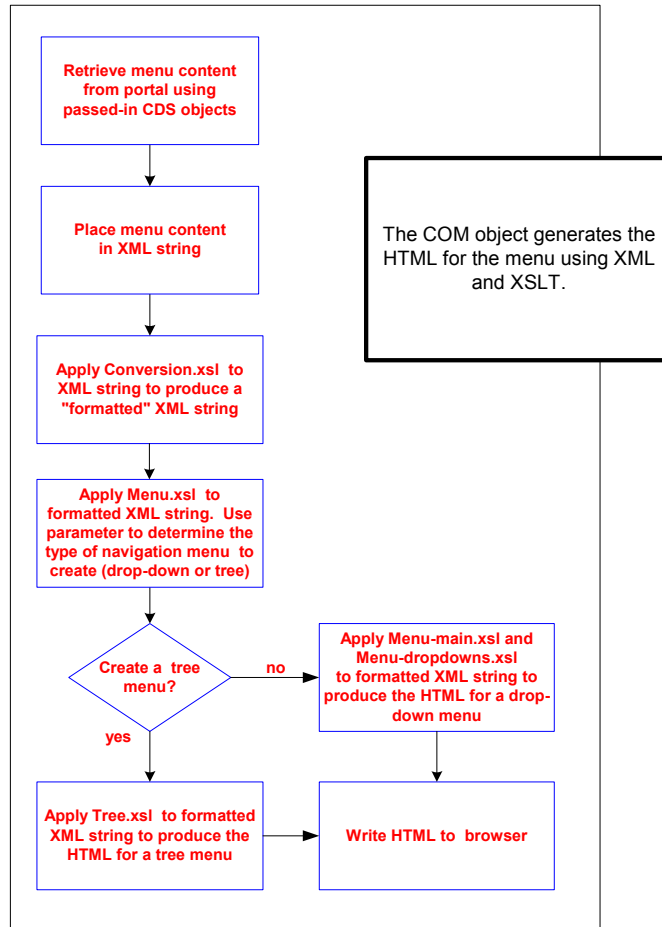
Menu.xsl imports the following XSL files:

- menu-main.xsl – generates the HTML for the header bar of a drop-down menu
- menu-spacers.xsl – generates the HTML for the spacer section of a drop-down menu. The spacer section is located between the header bar and drop-down lists.
- menu-dropdowns.xsl – generates the HTML for the drop-down lists of a drop-down menu
- tree.xsl – generates the HTML for a tree menu
- menu-scripts.xsl – generates the JavaScript section used by both the drop-down and tree menus.

Menu CDA Process Flow

The diagram below illustrates the process flow inside the COM object.

COM Object



Detailed Menu CDA Architecture

This section provides a detailed look at the Menu CDA Architecture. It is broken into sections describing the process of how the navigation menu is generated. These sections are listed below:

1. Menu CDA is executed
2. GenerateMenu() method is executed
 - a. CDA Caching is disabled
 - b. CDA Header is hidden
 - c. Menu content is retrieved and placed in XML string
 - d. (Optional) Menu content is cached in the Session
 - e. Conversion.xsl is applied to XML string
 - f. Menu.xsl is applied to Formatted XML string
 - g. HTML is written to browser

Menu CDA is Executed

When the Menu CDA is executed, it calls a COM object (CTXCDANavigation.NavigationServer) to generate and display the navigation menu for the portal. This COM object contains a method called GenerateMenu() that the Menu CDA calls to create the menu.

The GenerateMenu() method uses a series of passed-in CDS objects, such as Application, Presentation, and Personalize2, to retrieve the menu content from the portal. An XML string is also passed into the GenerateMenu() method to specify the names and locations of the XSL files used by the COM object to generate the HTML for the menu. This XML string also contains a value indicating whether or not to store all or parts of the generated menu content in the user's Session.

The GenerateMenu() method has the following interface:

```
Public Function GenerateMenu(strXMLParameters As String, objApplication, objHeader, _  
                             objPresentation, objPersonalize2, objRequest, objResponse, _  
                             objSession, Optional bDebug As Variant) As Boolean
```

The parameters of the GenerateMenu() method are discussed in the following table.

Table 1 - Parameters (of GenerateMenu)	
strXMLParameters	<p>(Required) XML string specifying the names of the two XSL files that the COM object uses to generate the menu.</p> <p>The XML string is constructed within the CDA using SAX (Simple API for XML) and has the following format:</p> <pre><menu-parameters> <configuration> <xml-transform name="conversion.xml" relative-path="true"/> <html-transform name="menu.xml" relative-path="true"/> <cache-type value="ica"> </configuration> </menu-parameters></pre> <p>The XML string above contains the following elements:</p> <ul style="list-style-type: none"> • xml-transform – contains the name and location of the Conversion.xml file. • html-transform – contains the name and location of the Menu.xml file • cache-type – indicates what menu content is “cached” in the user’s Session. Three different values can be specified: full, ica, and none. See section 2D of this document for more details on these cache types. <p>For both the xml-transform and html-transform elements, the “relative-path” attribute specifies how the “name” attribute is handled.</p> <p>When “relative-path” is true, the name attribute specifies a path to the XSL file relative to the location of the DLL file on the Agent Server. For example, in the XML string above, setting relative-path=”true” and name=”conversion.xml” indicates that the XSL file and DLL reside in the same folder.</p> <p>When “relative-path” is false, the name attribute specifies an absolute path to the location of the XSL file on the Agent Server. For example, if relative-path=”false”, the name attribute needs to contain a value such as “c:\program files\...\conversion.xml”.</p>
objApplication	<p>(Required) CDS Application object. Used to...</p> <ul style="list-style-type: none"> • Retrieve application variables such as pagepath, hostpath, category, page, and cdaid. These variables are used to construct the URLs for the menu items
objHeader	<p>(Required) CDS Header object. Used to...</p> <ul style="list-style-type: none"> • Hide the header for the Menu CDA

objPresentation	<p>(Required) CDS Presentation object. Used to...</p> <ul style="list-style-type: none"> Retrieve the list of folders and pages that the portal user has access to Retrieve the navigation type of the portal (tree or drop-down list)
objPersonalize2	<p>(Required) CDS Personalize2 object. Used to...</p> <ul style="list-style-type: none"> Retrieve the list of themes and “addable” cdas
objRequest	<p>(Required) CDS Request object. Used to...</p> <ul style="list-style-type: none"> Retrieve the portal name (configspace) and session id.
objResponse	<p>(Required) CDS Response object. Used to...</p> <ul style="list-style-type: none"> Turn off CDA caching by setting the CacheDuration property to 0. Write the HTML for the navigation menu to the browser.
objSession	<p>(Required) CDS Session object. Used to...</p> <ul style="list-style-type: none"> Retrieve session variables such as the username, password, and domain of the logged on user. These credentials are used to retrieve the ICA application list for the user. Save menu content to the user’s Session.
bDebug	<p>(Optional) Boolean indicating whether or not to create a Debug text file. The Debug text file contains all XML and HTML used and generated by the COM object.</p> <p>If True, a Debug text file is generated. The text file is placed in the same folder as the DLL on the Agent Server (Program Files\Citrix\Nfuse Elite\CDA\Agent\sbMenu).</p> <p>If False, no Debug text file is generated.</p> <p>By default, no argument is specified indicating that no Debug text file will be created.</p>

GenerateMenu() method is Executed

The GenerateMenu() is called by the CDA to create the navigation menu for the portal. The following steps are performed within the GenerateMenu() method.

CDA Caching is Disabled

Caching for the Menu CDA is disabled using the statement `Response.CacheDuration = 0`.

Note: This step indicates that “normal” CDA caching for the Menu CDA is disabled. A later step indicates that the Menu CDA may store all or parts of the menu content within the user’s Session. This menu content is “cached” within the Session, but the entire CDA is not being cached. This distinction should be noted.

CDA Header is Hidden

The Header for Menu CDA is hidden using the statement `Header.Visible = False`.

Menu content is retrieved and placed in XML string

The content for the menu is retrieved using the passed in CDS objects. The following content is retrieved from these objects:

- All folders and pages that will be available from the navigation menu. For a folder and page to appear in this list, the folder/page needs to be assigned to the user’s role from within the PMC.
- List of CDAs that can be added to a page from the navigation menu. For a CDA to appear in this list, the CDA must be designated as “addable to the Content Menu” from within...
 - its CDA properties (from within CDAPad)
 - the Role properties (from within the PMC)
- List of ICA applications that can be launched from the navigation menu. For ICA applications to appear in this list, the portal must be configured with a MetaFrame server from within the PMC.
- List of themes available in the portal. The theme list is not a standard menu item. It is only retrieved here to allow a developer to add the theme list to the menu if desired.

All menu content retrieved from the CDS objects is placed in an XML string. This XML string contains a raw form of the menu data, and is referred to as the “raw” XML string. Folders, pages, themes, CDAs, and ICA applications are grouped together, but are not arranged or listed alphabetically.

See Appendix B for the format of the raw XML string generated within this step.

(Optional) Menu content is cached in the Session

After the menu content is retrieved and placed within an XML string, all or parts of the XML string may be cached within the user's Session. The GenerateMenu() method uses the parameter, strXMLParameters, to determine if and how menu content should be cached within the user's Session. The strXMLParameters argument is an XML string containing various settings, among them the cache type of the CDA. (See Table1 above for more information on the structure and content of the strXMLParameters argument).

The Menu CDA can specify three different cache types for the menu to use: full, ica, or none.

- **“full”** cache type:
 - The Menu CDA caches the entire XML string containing menu content within the user's Session.
 - The first time the user accesses the portal, the menu content is generated and the XML string (containing menu content) is placed within the Session.
 - Subsequent calls to the portal retrieve the XML string from the Session directly, so that the content does not have to be regenerated.
 - Provides better performance when there is a low concurrent session count and/or a lot of RAM on the State Server with a fast network.
- **“ica”** cache type:
 - The Menu CDA caches the ICA application list section of the XML string within the user's Session.
 - The first time the user accesses the portal, all menu content (including the ICA app list) is generated.
 - The ICA app list section of the XML string is then stored in the user's Session.
 - When a subsequent call to the portal is made, the ICA app list section is retrieved from the Session, and all other menu content is regenerated.
 - Default setting for the Menu CDA
- **“none”** cache type:
 - The Menu CDA does not cache any menu content within the user's Session.
 - Offers the slowest performance since each page refresh forces all menu content to be gathered again.

Menu content is stored in the session under the value CTXNAVIGATION. When “full” caching is used, this Session variable contains an XML string containing all menu content. When “ica” caching is used, this Session variable contains an XML string containing only the ICA application list. When “none” caching is used, this Session variable doesn't contain a value. The contents of CTXNAVIGATION can be viewed by opening the Session text file on the State Server. This file is located at:

“Program Files\Citrix\NFuse Elite\Config\State Server\[portal_name]\Session\[letterA]\[letterB] where **portal_name** is the name of the portal and **letterA** and **letterB** are the first two letters of the text file.

Conversion.xsl is applied to XML string

After the XML string is generated (or retrieved from the user's Session), the Conversion.xsl file is used to transform the XML string into a formatted XML string used by the Menu.xsl file. It sorts the folders, pages, and CDAs of the raw XML string in alphabetical order and places menu content into "left" and "right" sides.

The Conversion.xsl transformation is used to separate the logistics of the menu from the presentation. It organizes the menu content based on where the menu items are accessed. This organization eases the manner in which the HTML output for the menu is produced from within the Menu.xsl transformation.

See Appendix C for the format for the XML string outputted by the Conversion.xsl transformation.

Menu.xsl is applied to Formatted XML string

After the Conversion.xsl file converts the raw XML string into a formatted XML string, the Menu.xsl file is applied to the formatted XML string. The Menu.xsl file is used to convert the formatted XML string into HTML.

The COM object passes a list of parameters to the Menu.xsl file to provide system information. The parameters that are passed in are listed below:

- **current-folder** – value of Application("CATEGORY") indicating the current folder the user is viewing.
- **current-page** – value of Application("PAGE") indicating the current page the user is viewing.
- **page-path** – value of Application("PATHPATH"). Used by the URLs of the menu items.
- **host-path** – value of Application("HOSTPATH"). Used by the URLs of the menu items.
- **page-full** – value of Personalize2.CurrentPageFull. Boolean indicating whether the current page has reached its max CDA limit.
- **page-personalizable** – value of Personalize2.CurrentPagePersonalizable. Boolean indicating whether the current page can be personalized (ie. CDAs can be added to the page or moved)
- **page-navigation** – value of Presentation.Property("Navigation"). String indicating whether the portal uses drop-down or tree navigation.
- **enable-icons** – Boolean indicating whether to show icons next to all menu items (default is True). This setting is made within the "Settings" option of the navigation menu.
- **cda-name** – name of the Menu CDA (sbMenu)
- **panel-folders** – number of folders to display in the Menu Header bar (default is 4). This setting is made within the "Settings" option of the navigation menu.
- **language** – XML file language (en-US)

The Menu.xsl file uses the above parameters to decide how the HTML for the menu is presented. The [page-navigation](#) parameter is used to decide the type of navigation used by the Menu. If its value equals "left", the Menu CDA uses left (or tree) navigation. If its value equals "top", the Menu CDA uses top (or drop-down) navigation.

Generating the tree menu

When tree navigation is specified, the Menu.xsl file applies the Tree.xsl file to the inputted XML string. The Tree.xsl file generates all HTML for the navigation tree.

Generating the drop-down menu

When drop-down navigation is specified, the Menu.xsl file applies the **Menu-main.xsl**, **Menu-spacers.xsl**, and **Menu-dropdowns.xsl** files to the inputted XML string. These XSL files are summarized below:

- **Menu-main.xsl** – generates the HTML for the header bar of the menu.
- **Menu-spacers.xsl** – generates the HTML for the spacer section located between the header bar and dropdown lists.
- **Menu-dropdowns.xsl** – generates the HTML for the dropdown lists of the menu.

Both the tree and drop-down navigation menus apply an additional XSL file called **menu-scripts.xsl**. This XSL file generates and displays the JavaScript used by the menus to the browser. The majority of JavaScript used by the menus is contained within a JavaScript include file called **menu.js**. Menu.js contains a generic set of JavaScript functions used for showing/hiding the menu elements. Menu.js is located on the Web Server inside the “[inetpub\wwwroot\\[portal_name\]\cds\sbMenu](#)” folder.

HTML is written to browser

After the HTML for the drop-down or tree menu is generated, it is written the browser.

Customizing the Menu CDA

Since the Menu CDA uses an open architecture, it can be customized to meet the needs of any business. Most alterations to the Menu CDA need to be done within the XSL files that generate the HTML for the menu.

Changes to the look and feel of the Tree menu need to be done within the Tree.xsl file. Tree.xsl generates all HTML used by the tree menu. This file is located on the Agent Server at “[Program Files\Citrix\NFuse Elite\CDA\Agent\sbMenu](#)”.

Changes to the look and feel of the Drop-down menu need to be done within the Menu-main.xsl, Menu-spacers.xsl, and Menu-dropdowns.xsl files. Menu-main.xsl generates the HTML for the menu header bar. Menu-spacers.xsl generates the HTML for the spacer section located between the header bar and drop down menus. Menu-dropdowns.xsl generates the HTML for the drop-down menus. These files are also located on the Agent Server at “[Program Files\Citrix\NFuse Elite\CDA\Agent\sbMenu](#)”.

Changes to the JavaScript code used by the both the drop-down and tree menu need to be done within the Menu-scripts.xsl file and Menu.js file. Menu-scripts.xsl generates the JavaScript code used by both menus. Part of the JavaScript generated by Menu-scripts.xsl is the inclusion of an external JavaScript file called Menu.js. Menu.js contains a series of generic functions used by both navigation menu types. Menu-scripts.xsl is located on the Agent Server at “[Program Files\Citrix\NFuse Elite\CDA\Agent\sbMenu](#)”. Menu.js is located on the Web Server at “[Inetpub\wwwroot\\[portal_name\]\cda\sbMenu](#)”.

A few ways to customize the Menu CDA are listed below:

- Add “Logout” as a link on the menu header bar
 - The logout button for a standard portal resides in the portal header. The logout feature can be moved to the menu header bar.
- Add “Home” as a link on the menu header bar
 - The “home” link for a standard portal resides under the My Pages menu option. The home link can be moved to the menu header bar for direct access.
- Add a custom pop-up link to the menu header bar
 - Additional items can be added to the menu header bar, including links that pop up browser windows with additional content.
- Provide theme change ability within the menu
 - To change a theme in a standard portal, a user needs to access the Settings page from the My Pages menu option. As an alternative, the theme list can be added to the menu to allow users to change themes in a more direct manner.
- Alter cache behavior used by the menu.
 - By default, the Menu CDA only caches the ICA application list used by the menu within the user’s Session. This can be changed to cache all menu content (or no menu content) within the user’s Session.
- Add show/hide functionality to the Tree Menu

- When the portal uses Tree navigation, the tree menu takes up a good portion of the page. Show and hide functionality can be added to display the tree menu when it is needed and minimize the tree when it is not needed.

To provide an example of how to implement a change to the Menu CDA, the following section outlines the steps needed to implement show/hide functionality to the Tree Menu.

Adding Show/Hide functionality to the Tree Menu

To add show/hide functionality to the Tree Menu, follow these steps:

1. Open the Tree.xml file. This file is located on the Agent Server inside the "Program Files\Citrix\NFuse Elite\CDA\Agent\sbMenu" folder.
2. Update the Tree.xml file by performing the following.

Find the section that starts with this in Tree.xml:

```
<xsl:template name="tree-main">
    <xsl:call-template name="scripts"/>
    <span class="CTXNav" style="height:100%;">
```

And add these two lines:

```
<span id="ctTreeLeft" style="cursor:hand; display:block; margin-bottom:3px;">
</span>

<span id="ctTreeRight" style="cursor:hand; display:none; margin-bottom:3px;">
</span>
```

Then modify the next line to have display on by default:

```
<table cellpadding="0" height="100%" class="tree" id="tree" style="display:block;">
```

3. Save and close the Tree.xml file.
4. Stop and restart all NFE services to view the change within the browser.

Debugging the Menu CDA

Generating a Debug text file

The COM object used by the Menu CDA provides the capability of generating a debug text file to track the progress of creating the menu. By default, the debug text file is not created when the Menu CDA is executed. However, if developers are customizing the menu CDA and would like to track various stages of the menu creation process, a flag can be set within the CDA to generate the debug text file.

The Debug text file contains the following data:

Parameters XML string – the XML string that is passed in as the first argument of the GenerateMenu() method. This XML string contains the names of the two XSL files used by the COM object, as well as the cache type for the CDA to use.

CDA Cache Type – the “cache” type used by the CDA. Has a value of full, ica, or none. This value is specified within the Parameters XML string that is passed into the COM object.

Menu XML string – the XML string containing raw menu content.

Transformed XML string – the XML string outputted by the Conversion.xsl transformation

Transformed HTML string – the HTML string that is written to the browser after the Menu.xsl transformation has been applied

Execution time – the execution times of various stages of the menu creation process are logged within the text file. All times are recorded in milliseconds.

To create the debug text file, follow these steps:

1. Open CDA Pad.
2. Select View->Framework CDAs from the file menu to get access to the NFuse Elite framework CDAs.
3. Select the sbMenu CDA from the CDA list.
4. In the Default action, change the call to the GenerateMenu() method to include a value of “True” as the last parameter. This last parameter is an optional argument that informs the COM Object to generate the Debug text file.

```
Call the object
If objNavigation.GenerateMenu(xmlWriter.output, Application, Header,
    Presentation, Personalize2, Request, Response, Session, True) = False Then
    Write out an error
    Response.CacheDuration = 0
    Response.Write ERR_TEXT
End If
```

5. Deploy the sbMenu CDA.
6. Launch the portal inside a browser window.

7. Log into portal. (The sbMenu CDA is re-executed and the debug text file is created)
8. View the Debug text file.

This file is located on the Agent Server at “**Program Files\Citrix\NFuse Elite\CDAs\Agent\sbMenu**” and the filename begins with “debug-”.

Note: A new debug text file is created each time a call to the portal is made.

Appendix

Summary of Menu CDA components

sbMenu CDA

The sbMenu CDA is part of the NFuse Elite framework. It calls a COM object to generate and display the navigation menu for the portal.

The sbMenu CDA code may need to be altered when:

- A developer wishes to generate a debug text file to view the XML and HTML used by the Menu CDA. See the “Debugging the Menu CDA” section for more information.
- The cache type used by the CDA needs to be changed. See section 2D of this document for the list and description of the different caching types.

COM DLL (CTXCDANavigation.dll)

The COM DLL is used by the Menu CDA to retrieve the menu content from the portal and generate the HTML for the menu. The COM object retrieves the content for the menu using a series of CDS objects that are passed into the COM object. It places the menu content in an XML string and applies a series of XSL transformations to convert the XML into HTML.

The COM DLL does not have to be altered when making changes to the look-and-feel of the navigation menu.

Conversion.xsl file

The Conversion.xsl file is the first transformation applied to the XML string that is generated by the COM object. This transformation converts the inputted XML into a new XML string. It organizes the XML into how the navigation menu will be presented, such as by grouping elements together and listing elements within a group in alphabetical order.

The Conversion.xsl file may need to be altered when:

- The organization performed by this transformation needs to be changed.
- The theme list needs to be included as part of the menu. The XML string inputted into this transformation contains the list of themes available within the portal. Conversion.xsl, however, does not output the theme list as part of the transformed XML. To add the theme list to the menu, Conversion.xsl needs to output the theme list as part of the outputted XML, and the XSL transformations applied later need to create the HTML for the theme list.

Menu.xsl file

The Menu.xsl file is the transformation applied immediately after the Conversion.xsl transformation is applied. The main responsibility of this file is to “direct” how the inputted XML string is converted into HTML. The Menu.xsl file imports a series of additional XSL files to actually perform the HTML conversion.

A series of parameters are passed into Menu.xsl containing system information. One of these parameters indicates the type of menu (tree or drop-down) that needs to be created. If a tree menu is to be created, Menu.xsl applies the Tree.xsl transformation to the inputted XML string. If a drop-down menu is to be created, Menu.xsl applies the menu-main.xsl, menu-spacers.xsl, and menu-dropdowns.xsl transformations to the inputted XML string.

The Menu.xsl file may need to be altered when:

A custom XSL transformation has been created to take the place of any of the XSL files imported into Menu.xsl. Menu.xsl needs to import and point to the new XSL transformation so that it can be used.

Tree.xsl

The Tree.xsl file generates all HTML for the tree menu. The tree menu consists of two levels of nodes. The parent nodes (or main nodes) of the tree consist of all folders that are accessible to the user, as well as “Add”, “Launch”, and “My Pages”. The child nodes of the tree consist of all content relating to the parent node. For example, the child node of a folder is the list of pages contained inside that folder. A parent node and its associated child node form a “node group”.

The tree menu is constructed using a series of embedded HTML tables. Each node group is contained within a separate table. The first row of the table contains the parent node. The second row of the table contains the child node. When a parent row is clicked within the browser, the child row is either displayed or hidden depending upon its current state.

The HTML for a tree menu can be viewed by displaying the source of the browser window (Select View->Source from the file menu)

The Tree.xsl file may need to be altered when:

- Any change to the look-and-feel of the tree menu needs to be incorporated

Menu-main.xsl

The Menu-main.xsl file generates the HTML for the header bar of the drop-down menu. The menu header bar displays all major options available within the menu. These options include a list of folder names, as well as “Add”, “Launch”, and “My Pages”. When an option is clicked, a drop down list is displayed containing related option items. The HTML for the drop-down lists is created within the Menu-dropdowns.xsl file.

The Menu-main.xsl file may need to be altered when:

- Any change to the look-and-feel of the menu header bar needs to be incorporated

Menu-spacers.xsl

The Menu-spacers.xsl file generates the HTML for the spacer section of the drop-down menu. The spacer section is located between the header bar and drop-down lists. Spacing between the menu header bar and drop-down menu is controlled by a spacer image.

The Menu-spacers.xsl file may need to be altered when:

- A change to the spacer image size needs to be incorporated

Menu-dropdowns.xsl

The Menu-dropdowns.xsl file generates the HTML for the drop-down lists of the drop-down menu. The drop-down lists are displayed when a user clicks an option in the menu header bar.

All drop-down lists are HTML tables that are set to be visible or hidden. When an option in the menu header bar is clicked, the drop-down list associated with the header bar option is set to be visible, and the previous drop-down list that was displayed is hidden. All HTML used by the navigation menu can be viewed by displaying the source of the browser window (Select View->Source from the file menu)

The Menu-dropdowns.xsl file may need to be altered when:

- Any change to the look-and-feel of the drop-down menus needs to be incorporated

Menu-scripts.xsl

The Menu-scripts.xsl file generates the JavaScript section used by both the drop-down and tree navigation menus. Most of the JavaScript used by the menus is included within an imported JavaScript file called Menu.js. Menu.js contains functions to show and hide the drop-down lists.

The Menu-scripts.xsl file or Menu.js file may need to be altered when:

- Any change to the client-script used by the navigation menus needs to be incorporated

XML retrieved from CDS objects

The XML string retrieved from the CDS objects has the following format:

```
<menu-data>
  <mypages-list>
    <folder name=imy pagesï friendly-name=ïMy Pagesï>
      <page name=ïï friendly-name=ïHomeï url-prefix=ïhost-pathï url=ïï/>
      <page name=ïmypageï friendly-name=ïMy Pageï url-prefix=ïhost-pathï
        url=ï?category=&page=mypageï/>
      <page name=ïsettingsï friendly-name=ïSettingsï url-prefix=ïhost-pathï
        url=ï?category=&page=settingsï/>
```

```

    </folder>
</mypages-list>
<folder-list>
  <folder name=ïfolder1ï friendly-name=ïFolder1ï>
    <page name=ïpage1ï friendly-name=ïpage1ï url-prefix=ïhost-pathï
      url=ï?category=folder1&amp;page=page1ï/>
    <page name=ïpage2ï friendly-name=ïpage2ï url-prefix=ïhost-pathï
      url=ï?category=folder1&amp;page=page2ï/>
  </folder>
</folder-list>
<themes-list>
  <theme friendly-name=ïChromeï url-prefix=ïpage-pathï
    url=ï&amp;cmd=changetheme&amp;themeid={68575A3A-Ö} | Chromeï/>
</themes-list>
<cda-list>
  <cda-application friendly-name=ïWebsite Viewerï url-prefix=ïpage-pathï
    url=ï&amp;cmd=addcda&amp;cdaid={A5DBC0E4-Ö} | Website Viewerï/>
</cda-list>
<ica-list>
  <ica-application friendly-name="Notepad" description="Notepad on server"
    icon-small-url="http://augusta/myportal/cds/host.xps/getSmallIconURL/Notepad"
    icon-large-url="http://augusta/myportal/cds/host.xps/getIconURL/Notepad"
    content="false" embed-url="http://augusta/myportal/cds/host.xps?page=test1
      &category=folder%201&rqid=pklycl&cmd=addcda&cdaid={A5DBC0E4-} | Embedded
      Application&appname=Notepad&friendlyname=Notepad"
    launch-url="http://augusta/myportal/cds/host.xps?category=&page=NfuseLaunch
      &action=launchica&cmd=refreshraw&cdaid={A5DBC0E4-Ö} | NfuseLaunch
      &cdainstanceid=CDA67AE1Ö&method=SEAMLESS&appname=Notepad" />
</ica-list>
</menu-data>

```

The XML string above contains the following elements:

- **menu-data** – root node of the XML
- **mypages-list** – contains content that is available under the “My Pages” section of the menu

- **folder-list** – contains list of folders and pages that will be accessible from the menu
- **themes-list** – contains list of themes that can be applied to the portal. The theme can be changed under the “Settings” option of the “My Pages” section.
- **cda-list** – contains list of CDAs that can be added to a page from the content menu. This list is maintained within the PMC inside the Role Properties dialog.
- **ica-list** – contains list of ICA applications that will be accessible from the content menu

XML outputted by Transformation.xsl

The XML string outputted by the Conversion.xsl transformation has the following format:

```
<navigation>
  <folder-group align=iflefti>
    <folder friendly-name=ifFolder1i>
      <item icon-url=ifp.gifi friendly-name=ifpage1i href-prefix-code=if1i
        href=if?category=Folder1&amp;page=page1i/>
      <item icon-url=ifp.gifi friendly-name=ifpage2i href-prefix-code=if1i
        href=if?category=Folder1&amp;page=page2i/>
    </folder>
  </folder-group>
  <folder-group align=ifrighti>
    <folder friendly-name=ifAddi>
      <item friendly-name=ifWebsite Vieweri enabled=iffalsei
        href-prefix-code=if0i href=if&amp;cmd=addcda&amp;cdaid=
        {A5DBC0E4-Ö}|Website Vieweri icon-url-prefix-code=if2i
        icon-url=ifc.gifi/>
    </folder>
    <folder friendly-name=ifLaunchi>
      <item friendly-name="Notepad"
        href="http://localhost/myportal/cds/host.xps?category=&amp;
        page=NfuseLaunch&amp;action=launchica&amp;cmd=refreshraw&amp;
        cdaid={A5DBC0E4Ö}|NfuseLaunch&amp;cdainstanceid=CDA67AE1Ö
        &amp;method=SEAMLESS&amp;appname=Notepad"/>
    </folder>
    <folder friendly-name=ifMy Pagesi>
      <item icon-url=ifp.gifi friendly-name=ifHomei href-prefix-code=if1i href=if1i/>
```

```
<item icon-url=îp.gifî friendly-name=îMy Pageî href-prefix-code=î1î
  href=î?category=&page=mypageî/>
<item icon-url=îp.gifî friendly-name=îSettingsî href-prefix-code=î1î
  href=î?category=&page=settings&lastcategory=&
  lastpage=DefaultHomepageî/>
</folder>
</folder-group>
<prefixes>
  <prefix key=î1î value=îhttp://server1/portall/host.xpsî/>
  <prefix key=î0î value=îhttp://server1/portall/host.xps?page
    =DefaultHomepage&category=&rqid=x1r5e1î/>
  <prefix key=î2î value=îsbMenuî/>
</prefixes>
</navigation>
```

The XML string above contains the following elements:

- **navigation** – root node of the XML
- **folder-group align="left"** – contains list of folders and pages that will appear on the left side of the navigation menu
- **folder-group align="right"** – contains list of menu items that will appear on the right side of the navigation menu (such as Addable CDAs, ICA applications, and My Pages)
- **prefixes** – contains the URL prefix codes for each of the menu items
 - key = 0 corresponds to Application("PAGEPATH")
 - key = 1 corresponds to Application("HOSTPATH")
 - key = 2 corresponds to the CDA name



**851 West Cypress
Creek Road**

Fort Lauderdale, FL 33309

954-267-3000



<http://www.citrix.com>

Copyright © 2000 Citrix Systems, Inc. All rights reserved. Citrix, WinFrame and ICA are registered trademarks, and MultiWin and MetaFrame are trademarks of Citrix Systems, Inc. All other products and services are trademarks or service marks of their respective companies. Technical specifications and availability are subject to change without prior notice.