NetFlow for SouthWare NetLink™

NetFlow is a technology add-on for the SouthWare NetLink module that allows customization of NetLink web pages **without modifying the standard page templates or requests**! You can customize most features of SouthWare portals and web pages *while maintaining compatibility with future standard releases.*

Overview

SouthWare NetFlow technology utilizes a concept that is similar to SouthWare WorkFlow in order to provide "custom without custom" capability for SouthWare NetLink pages and portals. This allows users to create modifications that are stored outside of the standard NetLink request html templates similar to the way that WorkFlow stores modifications outside of the standard programs.

Comparison of Concepts:	**SouthWare WorkFlow™**	**SouthWare NetFlow™**
Object to Modify	Program	NetLink Request (template)
Aspect to Modify	FlowPoint	NetPoint or Document Object Model
- Where Defined	▪ In Program	▪ In Template
Modification	FlowMod	NetMod
- Where Stored	▪ In FlowMod file	▪ In NetMod file (and optional text files)

This concept should be easy to grasp for existing SouthWare users who are familiar with WorkFlow.

Benefits

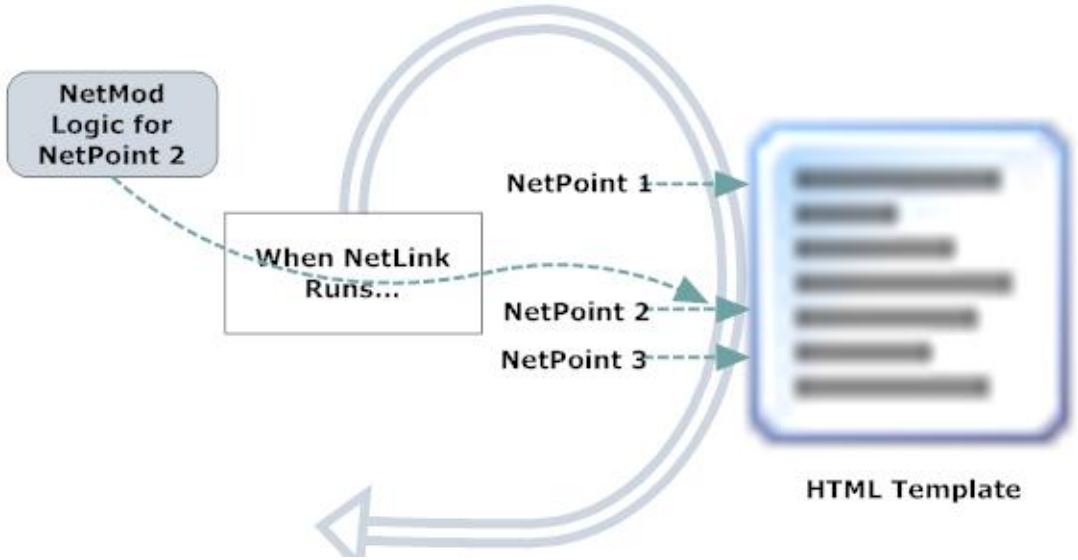
1. You can quickly create changes such as:
    - Hiding or adding content on standard pages
    - Replacing the HTML code in a section of the page
    - Adding a button to a toolbar
    - Adding new functions to a page
    - Modifying functions
    - Replacing labels for table rows
    - Adding rows to tables
    - Use alternative ReportMate or ImportMate formats
    - ….Modifying any element of the Document Object Model that has an ID
2. ***All of these changes can be done conditionally based on IF/AND/OR conditions!***
3. Your changes are safely stored in a separate file that is not affected by the loading of new templates
4. If your customization needs are more extensive than just modifying the template for a particular request, you can still use NetFlow to conditionally:

- Redirect all NetLink pages to go to an alternate request instead of the standard request
- Specify to use a different template for a request

5. You can quickly identify the custom changes that have been made on your system
6. You can implement NetFlow changes gradually while retaining backwards compatibility with existing customized pages

Contents

How It Works



When a NetLink request is processed, NetLink tests for the existence of any NetMods for that request.

- If a NetMod exists for NetPoint "*request" then NetLink will test whether to switch to the alternate request
- If a NetMod exists for NetPoint "*template" then NetLink will test whether to change the template for this request
- If a NetMod exists for NetPoint "*rmformat" then NetLink will process any alternate RM formats or alternate IDs instead of the standard ones
- If a NetMod exists for NetPoint "*imformat" then NetLink will process the alternate import format instead of the standard one
- For other NetPoints NetLink will test and SKIP or REPLACE logic as specified
- To add or modify functions you can insert code at a special NetPoint at the end of the header scripts
- To modify the page via the Document Object Model (DOM) you can insert code at a special NetPoint that is inserted as the page is loaded

Strategy Tips (and Why You Need to Use NetFlow)

The primary goal with NetFlow is to make it possible for you to have custom changes to SouthWare NetLink pages that will automatically remain in place when you load future SouthWare updates. It may seem simpler and quicker in the short run to just modify the standard template for a custom change. But when you do that you won't be able to load a future SouthWare replacement for that page without the need to modify the standard page again (assuming you remember what you changed☺). This can be

very tedious and frustrating work, particularly if the custom changes are not well-documented in the page.

With a little thought and the use of NetFlow you can confidently load future updates AND have your custom too. Here are some tips on the most common changes.

When you need to modify the body of the page:

The best way to do this is via modifying the Document Object Model (DOM) at the "ONLOAD_ADD" NetPoint. The NetPoint doesn't have any standard code, so your custom code won't interfere with standard code. After the page is loaded you can reference elements in the body via their "id" to hide or change them. You can also add rows to tables and other elements via the DOM. See the samples page for more details.

When you need to modify some container or navigation elements of the page (tabs, layout, menus, toolbars, etc.)

Typically these elements are defined in the script section at the end of the page, so the best way to modify these is also via the "ONLOAD_ADD" NetPoint. There are also special NetPoints to add tabs and toolbars since there are multiple steps involved in adding new options. See the sample page for more details.

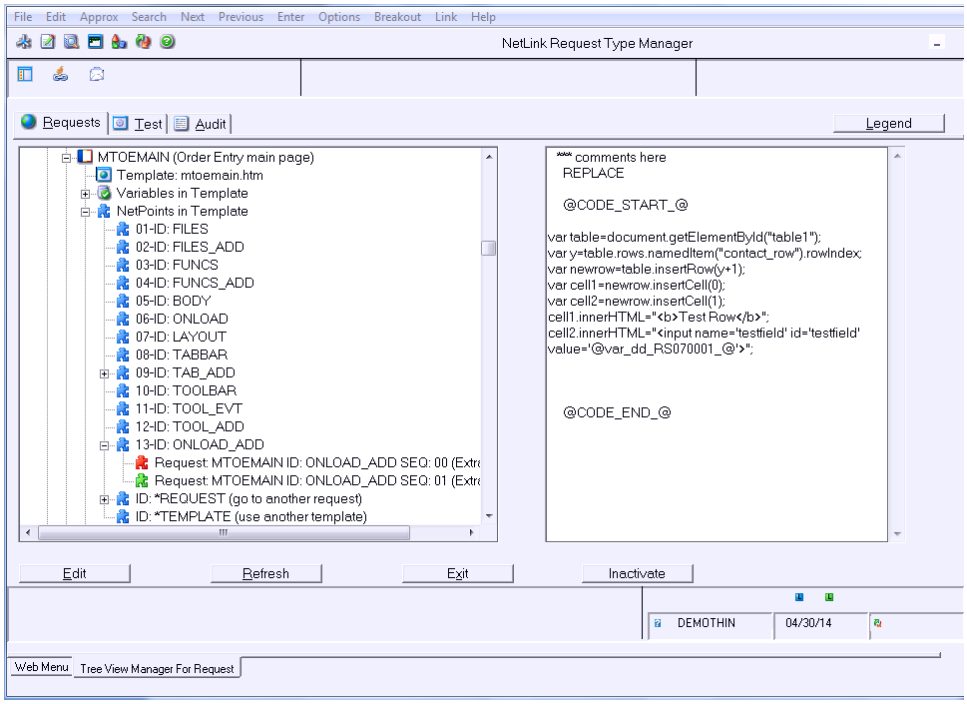
When you need to add or modify a function within the page

The functions defined in the header section of the page can be modified by replacing the contents of the "FUNCS" NetPoint, but it is better to use the "FUNCS_ADD" NetPoint if possible. Via the "FUNCS_ADD" NetPoint you can add new functions and replace the code of existing functions. This is preferable to replacing the "FUNCS" section because SouthWare might add future functions to the page that would not be in the FUNCS code you insert via your saved NetMod. If you use the "FUNCS_ADD" approach you will get the benefit of any new functions plus your custom changes. See the sample page for more details.

When you need to modify the request, RM formats, or IM format

Use the special NetPoints that let you modify these features instead of copying them to new records that will then be "locked in time". See the sample page for examples.

NetLink Manager



The NetLink Manager program provides access to views and editing of the NetFlow information.

1. Expansion of a request includes a “NetPoints in Template” option which shows all NetPoints found in the template
2. The list of NetPoints can be expanded to show any NetMods that are defined
3. If you highlight a NetMod in the manager the preview displays the NetMod logic
4. You can add a NetMod for a NetPoint or edit an existing NetMod
5. A “View All NetMods” section of the tree lists all NetMods that are defined
6. You can quickly toggle a NetMod to be Active/Inactive when the NetMod is highlighted – this can help when debugging custom changes

NetPoints

A NetPoint is a section of an html template that is surrounded by comments that identify that section as a NetPoint. NetPoints are defined similar to the conditional comments feature already available in NetLink (the existing conditional features will still apply and will override any NetMods if applicable).

A NetPoint (10 character maximum for ID) can be added to any template by simply surrounding a section with the following syntax:

Syntax within the body:

<!—@conditional_start:netpoint=*xxxNetPointIDxxxxx*_@ -->
(section of code)
<!-- @conditional_end_@ -->

Syntax within a script:

//<!—@netpoint_start:id=*NetPointIDx;desc=xxx30-char-desc*_@ -->
(section of script)
//<!-- @netpoint_end:id=*NetPointIDx*_@ -->

Standard NetPoints that may be available in a page:

NetPoint ID	Position	Description
FILES	In header	Allows replacement of section of page that references script files and links to css files <!-- @netpoint_start:id=FILES;desc=All Header Include Files_@ --> <!-- @netpoint_end:id=FILES_@ -->
FILES_ADD	In header	To add a new include file without replacing others <!-- @netpoint_start:id=FILES_ADD;desc=Add Include Files_@ --> <!-- @netpoint_end:id=FILES_ADD_@ -->
FUNCS	Within script tag in header	Allows replacement of all header functions //<!-- @netpoint_start:id=FUNCS;desc=All Header Scripts_@ --> //<!-- @netpoint_end:id=FUNCS_@ -->

NetPoint ID	Position	Description
FUNCS_ADD	At end of script section in header	To add a new function to header of page – you can also use this section to overwrite an existing standard function //<!-- @netpoint_start:id=FUNCS_ADD;desc=Add function in header_@ --> //<!-- @netpoint_end:id=FUNCS_ADD_@ -->
BODY	At start/end of body	To replace entire body of page <!-- @netpoint_start:id=BODY;desc=Entire body of page_@ --> <!-- @netpoint_end:id=BODY_@ -->
(IDs within body)	In body	These are not NetPoints, but they allow altering the body of the page by referencing the IDs via script lines inserted in the ZADDEND NetPoint. The following elements typically have IDs: • div tag • span tag • table tag • table row tag • buttons • links (anchor tags) • forms • form input fields
ONLOAD	Within script tag at end of body	To replace entire script section that is executed when page loads //<!-- @netpoint_start:id=ONLOAD;desc=All script at end of page_@ --> //<!-- @netpoint_end:id=ONLOAD_@ -->

ONLOAD_ADD	At end of script at end of body	To add new script lines to perform at end of loading page **– this is the NetPoint typically used to modify the DOM**
		
		//<!-- @netpoint_start:id=ONLOAD_ADD;desc=Add script at end of page_@ -->
		/*example syntax:
		// to hide an object with an id of AAA:
		// document.getElementById(“AAA”).style.display = “none”;
		// to replace the html within an object with id of BBB:
		// document.getElementById(“BBB”).innerHTML = newhtmlcode;
		*/
		//<!-- @netpoint_end:id=ONLOAD_ADD_@ -->

NetPoint ID	Position	Description
LAYOUT	Around layout definition script	To replace definition of layout (add _B, _C, etc. for multiple layouts) //<!-- @netpoint_start:id=LAYOUT;desc=Script for layout_@ --> //<!-- @netpoint_end:id=LAYOUT_@ -->
TABBAR	Around tabbar definition script	To replace definition of tabbar (add _B, _C, etc. for multiple layouts) //<!-- @netpoint_start:id=TABBAR;desc=Script for tabbar_@ --> //<!-- @netpoint_end:id=TABBAR_@ -->

TAB_ADD	At end of tabbar definition script	To add a new tab to a tabbar
		
		//<!-- @netpoint_start:id=TAB_ADD;desc=Area to add new tab_@ -->
		/*sample syntax:
		tabbar.addTab("details","S/O View","100px");
		tabbar.setContent("list","listarea");
		*/
		//<!-- @netpoint_end:id=TAB_ADD_@ -->

TOOLBAR	Around toolbar definition script	To replace definition of toolbar //<!-- @netpoint_start:id=TOOLBAR;desc=Script for toolbar_@ --> //<!-- @netpoint_end:id=TOOLBAR_@ -->

TOOL_ADD	At end of toolbar definition script	To add a new option to the end of a toolbar
		
		//<!-- @netpoint_start:id=TOOL_ADD;desc=Area to add toolbar option_@ -->
		/*sample syntax:
		toolbar.addButton("full", 1, "Full Entry", "images/fulledit.png");
		toolbar.setItemToolTip("full", "Go to Advanced Maintenance");
		*/
		//<!-- @netpoint_end:id=TOOL_ADD_@ -->

TOOL_EVT	Within options for toolbar onClick event	To add a new option to the onClick event for a toolbar //<!-- @netpoint_start:id=TOOL_EVT;desc=Area to add toolbar event_@ --> /*sample syntax: case "myoption": (myfunction to launch) break; */ //<!-- @netpoint_end:id=TOOL_EVT_@ -->

NetMods

A NetMod is a stored command or logic that may conditionally or always modify the standard processing of a NetLink request at a particular NetPoint. NetMods are stored outside of the request and template so that you may continue to load updated versions of the standard request/template while retaining the customized processing of your NetMods.

Generally a NetMod is designed so that you may perform conditional tests (IF, AND, etc.) and choose to Skip the output of the NetPoint (section) or Replace the section of code with other code. A NetMod can also be used to conditionally:

- switch processing to another request
- use an alternate template
- use a substitute ReportMate format or alternate ID
- use a substitute ImportMate format
- add or modify a function in the page
- modify any element of the Document Object Model that has an ID to reference

Examples:

IF (something)
    SKIP (don't output this section)

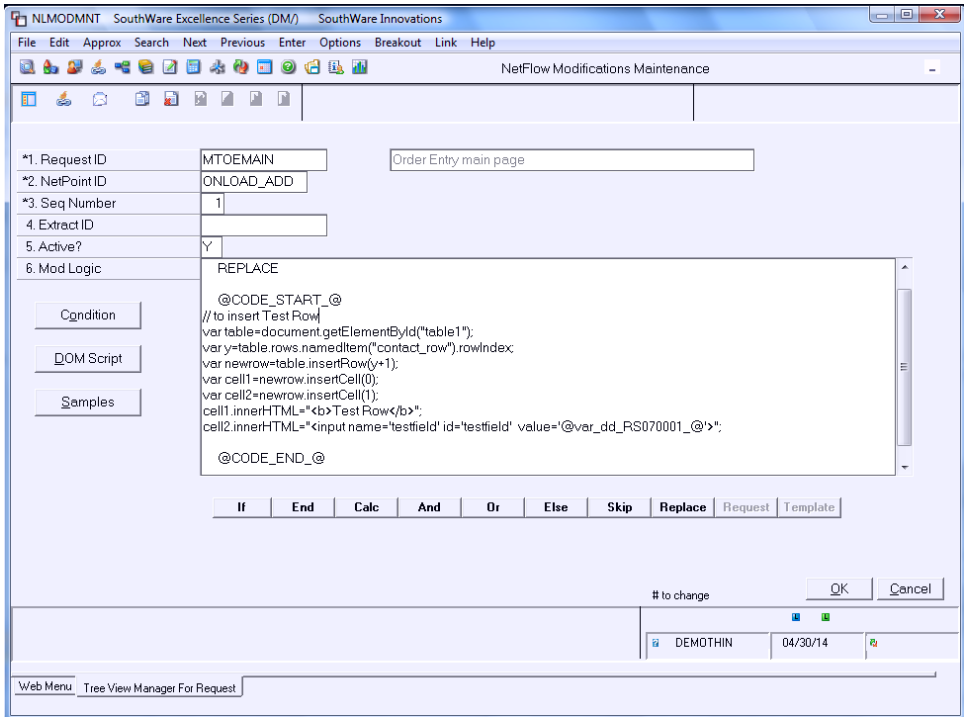
IF (something)
    REPLACE (provide alternate code)

IF (something)
    REQUEST (call a different request ID)

IF (something)
    TEMPLATE (use a different template)

IF(something)
    (do any scripting or DOM methods)

NetMod Maintenance



The setup and maintenance of NetMods is done via the NetMod maintenance program. The easiest way to setup or edit a NetMod is through the NetLink Type Manager program. When you expand a request in the tree one of the options is “NetPoints in Template”. When you expand this option the program reads the template and displays each NetPoint found in the template. You can then choose to zoom/edit a NetPoint to access the NetMod maintenance program for that NetPoint.

Request ID:

Enter the request ID to be modified.

NetPoint ID:

Enter the NetPoint of the section to be modified.

Seq Number:

Press [Enter] or enter a sequence number to uniquely identify this NetMod. The sequence number allows you to have multiple NetMods for the same NetPoint.

Extract ID:

Enter the Extract ID for this mod.

Active?:

Indicate whether this mod is currently active.

Mod Logic:

In this 5000-character text field you enter the modification logic, each step on a separate line. You may enter the logic directly or you may use the helper wizards described below under "Setup Wizards".

- Conditional Test Lines:
  You may enter a test in the format *(conditional) (value 1) (test) (value 2)*
  - where conditional is:
    - IF
    - AND
    - OR
  - and value 1 or 2 may be:
    - A constant number: @num_xxxxxxxxx_@
    - A literal: @lit_xxxxxxxxx_@
    - A date: @date_xxxxxxxx_@
    - A data dictionary field: @ddvar_XX999999_@
    - A NetLink variable: @nlvar_xxxxxxxxxxxx_@
  - and test may be:
    - EQ (equal to)
    - LT (less than)
    - GT (greater than)
    - NE (not equal to)
    - GE (greater than or equal to)
    - LE (less than or equal to)
    - MM (match masking – characters of value 1 are found in value 2)
    - NM (no match masking – characters of value 1 are no found in value 2)
- Conditional Modifiers:
  You may also enter one of the following to modify the test:
  - ELSE (to begin logic to be executed if the previous IF logic is not true)
  - END (to end an IF test)
- Calculation:
  You may create a calculation that you can then use in subsequent steps: You can use the CALC command to calculate a number and then use the number in a conditional test or use the number within a page modification. The syntax is:

  CALC operand1 operation operand2 = @calc_CALCn@ where:

- operand1 can be a data dictionary field, a NetLink variable, a previously calculated value
- operation can be:
  - "+" for add
  - "-" for subtract
  - "/" for divide
  - "*" for multiply
  - "%" for percentage of operand2
- operand2 can be a data dictionary field, a NetLink variable, a previously calculated value, or a constant
- CALCn is the result of the calculation where n is a number from 1 to 9 to uniquely identify this calculation result

For example, to calculate the difference between a customer's credit limit (AR010014) and their existing balance (AR010029) you could do a calculation such as:

CALC @ddvar_AR010014_@ - @ddvar_AR010029_@ = @calc_CALC1_@

You could then use @calc_CALC1_@ as a comparison value in a conditional test or as a javascript variable to modify the page.

- Actions to Take:
  You may enter one of the following actions for the NetMod:
  - SKIP (don't output this section)
  - REQUEST xxxrequestIDxxx (stop processing this request and call another request instead – available only at NetPoint "@request")
  - TEMPLATE xxxxxxxxxxxx (use this template for this request – available only at NetPoint "@template")
  - REPLACE (replace this section with alternate code/text from a file or from the following lines enclosed with @CODE_START_@ and @CODE_END_@)
    - @FILE=xxxxxxxxxxx (insert the contents of file xxxxxxxxx instead of the standard section – the file should exist in the template directories)
    - @CODE_START_@
      (line or lines of alternate code/text)
      @CODE_END_@
      - The Mod Logic field is limited to 5000 characters, so if your logic and alternate code/text exceeds this limit you will need to put the alternate text in a separate file and use the @FILE= method.
- Comments
  You may also enter comment lines in your logic to document the steps.

- If a line starts with an asterisk (*) it will be treated as a comment rather than as a step to process, and it will not be inserted into the template at run time.
- If the line starts with a forward slash (/) it is also treated as a comment, but it will be inserted into the template at run time.

"Condition" Wizard

In the Mod Logic field you may use the "Condition" wizard button to help you enter a conditional test. You can select the Type of Condition and the related operands and operator for a conditional test. This provides a quick and accurate way to enter the correct syntax for data dictionary fields, literals, dates, etc.

"DOM Script" Wizard

At the ONLOAD_ADD NetPoint an additional button "DOM Script" is available for the Mod Logic field. This button launches a wizard to help you create javascript commands that perform common manipulations of the Document Object Model. These script commands modify the page based on the IDs assigned to elements within the page.

Here are available options for this wizard at the time of this writing:

- **Hide Object, Remove Space –** this removes the display of the object and removes the space that the object occupies (changes style to "display:none")
  - You specify the ID of the object – you may press [F3] to display a list of IDs that the wizard finds in the template
  - The wizard will insert the code:

    document.getElementById("*id*").style.display = "none";

- **Hide Object, Leave Space** – this removes the display of the object but leaves the space that the object occupies (changes style to "visibility:hidden")
  - You specify the ID of the object – you may press [F3] to display a list of IDs that the wizard finds in the template
  - The wizard will insert the code:

    document.getElementById("*id*").style.visibility = "hidden";

- **Replace Object's HTML** – this replaces the HTML contained within the object (changes innerHTML)
  - You specify the ID of the object – you may press [F3] to display a list of IDs that the wizard finds in the template
  - The wizard will insert the code:

document.getElementById("*id*").innerHTML ="*MY NEW CODE*";

- You should replace the literal "MY NEW CODE" in the code with the appropriate HTML-formatted text.

- **Change Table Row Label** – this replaces the label of a row in a table that has IDs for the table and the row (modifies the cells/rows property for the table)
  - You specify the ID of the table – you may press [F3] to display a list of IDs that the wizard finds in the template
  - You specify the ID of the row – you may press [F3] to display a list of IDs that the wizard finds in the template
  - The wizard will insert:

    var x=document.getElementById("*tableid*").rows.namedItem("*rowid*").cells;
    x[0].innerHTML="*MY NEW LABEL*";

  - You should replace the literal "MY NEW LABEL" in the command line with the appropriate text.
- **Add New Table Row for a data field** – this inserts a new table row (with two columns) in a table that has IDs for the table and for the rows
  - You specify the ID of the table – you may press [F3] to display a list of IDs that the wizard finds in the template
  - You specify the ID of the row that will be immediately above the added row – you may press [F3] to display a list of IDs that the wizard finds in the template
  - The wizard will insert:

    var table=document.getElementById("*tableid*");
    var y=table.rows.namedItem("*rowid*").rowIndex;
    var newrow=table.insertRow(y+1);
    var cell1=newrow.insertCell(0);
    var cell2=newrow.insertCell(1);
    cell1.innerHTML= "My New Row Label";
    cell2.innerHTML="@var_dd_xxxxyyyy_@"; //where xxxxyyyy is a data dictionary field to display

  - You should replace the literal "My New Row Label" in the command line with the appropriate text for the label.
  - You should replace the xxxxyyyy with the appropriate data dictionary field

Samples/Examples:

***to skip the display of this section if operator is DEMO
IF @ddvar_XX990001_@ EQ @lit_DEMO_@
SKIP
END

***to replace the code in this section if the NetLink variable "MYTEST" equals Y
***and the customer past due balance is greater than 100
IF @nlvar_MYTEST_@ EQ @lit_Y_@
AND @ddvar_AR010045_@ GT @num_100_@
REPLACE
@CODE_START_@
code/text to use in this section
@CODE_END_@
END

For more extensive examples see the sample page available from the "Samples" button in the NetMod maintenance program (file name is nlmodsamples.htm). At the time of this writing this included samples for:

- Removing the Contents of a NetPoint
- Changing the Contents of a NetPoint
- Conditionally Removing/Changing a NetPoint
- Calculating a Value
- Calculate Difference Between Two Dates
- Calculate a Date
- Using a NetMod to Switch to an Alternate Request
- Using a NetMod to Switch to an Alternate Template
- Using a NetMod to Switch to Alternate ReportMate Format(s) for a Request
- Using a NetMod to Switch to ReportMate Alternate ID for a Request
- Using a NetMod to Remove ReportMate Format(s) for a Request
- Using a NetMod to Switch to an Alternate ImportMate Format
- Adding a New Include File (Script or CSS)
- Adding a New Function
- Replace/Modify a Function
- Adding a New Tab to a Tabbar
- Adding a New Element to a Toolbar
- Modify a Layout Configuration
- Modifying the Page using the Document Object Model (DOM)
- DOM - Hide an object via its ID and remove the space it occupies
- DOM - Hide an object via its ID without changing the space it occupies
- DOM - Replace the HTML in an object via its ID

- DOM - Hide a Graph via its ID
- DOM - Remove a Tree Menu Option
- DOM - Change Label of a Tree Menu Option
- DOM - Add a Tree Menu Option
- DOM - Move a Tree Menu Option
- DOM - Remove Tab from Tabbar
- DOM - Change Tab Label in Tabbar
- DOM - To change the label of a table row
- DOM - To CONDITIONALLY change the label of a table row
- DOM - To add a new row to a table