finPOWER Connect 3
Summary Pages (Version 2)

Version 3.03

5th June 2020

Table of Contents

Disclaimer

Version History

Date	Version	Name	Changes
07/08/2014	1.00	PH	Created.
28/11/2014	1.01	PH	Updates to HTML Show Application Shortcuts.
15/12/2014	1.02	PH	Typos fixed relating to AddDouble.
07/01/2015	1.03	PH	Icon HTML templates now support overlays.
12/01/2015	1.04	PH	SystemColour HTML template added.
20/01/2015	1.05	PH	Enhanced FormShow Application Shortcut for Create Document wizard.
02/02/2015	1.06	PH	Button templates added.
13/02/2015	1.07	PH	HtmlShowFunction Application Shortcut updated to accept 'popup' parameters.
25/03/2015	1.08	PH	Added Form Summary Text section.
04/05/2015	1.09	PH	Disclaimer template and CreditReport type summary pages.
01/07/2015	1.10	PH	New CreditEnquiry type Application Shortcut.
14/08/2015	1.11	PH	DocumentManagerImage HTML template.
24/08/2015	1.12	PH	New WebServicesToken type Application Shortcuts.
14/03/2016	3.00	PH	Updated for finPOWER Connect version 3.
11/07/2016	3.01	PH	Added "Action" HTML Template.
16/01/2017	3.02	PH	Address "Address" HTML Template.
05/06/2020	**3.03**	**PH**	**ResourceImage Template.**

Introduction

This document discusses finPOWER Connect Summary Pages, in particular, 'Summary Page (version 2)' type Scripts.

Prior to finPOWER Connect 2, code within Summary Page Scripts directly created HTML which could cause the following issues:

- Inconsistent formatting
    - Since the HTML could be specified directly, any HTML styles such as fonts and colours could be used which could lead to an inconsistent 'look and feel' between pages.
- Invalid HTML
    - Scripts generating HTML directly were prone to producing invalid HTML, E.g., tags may have been nested incorrectly and closing tags accidentally omitted.
- Internet Explorer specific HTML
    - Summary Pages within finPOWER Connect target the 'Web Browser' control which, behind the scenes, uses Microsoft Internet Explorer.
    - Early Summary Page Scripts targeted IE-only functionality such as specifying column alignment for tables in `<col>` tags.
        - Later versions of HTML supported by IE (e.g., HTML 5) no longer support this.

The introduction of 'Summary Page (version 2)' Scripts in finPOWER Connect 2 has the following advantages:

- No HTML
    - Summary Page code no longer uses HTML directly thereby removing all the HTML-based disadvantages of the original Summary Pages.
        - There may be the odd exception to this where HTML is used but most Summary Page Scripts never use HTML directly.
- Centralised customisation
    - In the original Summary Pages, if you wanted to update a block of information that appeared in more than one place, E.g., Account Monitoring which appears on both the Account Key Details and Status pages, you had to update two Scripts.
    - The concept of 'Standard Blocks' was introduced in finPOWER Connect 2 meaning that information can be updated centrally. This is detailed later in this document.
- Responsive styles
    - The HTML generated in version 2 Summary Pages can be responsive, i.e., it can react to different screen sizes, E.g.:
        - Hide table columns on small screens.

This document can be used as a reference when writing 'Summary Page (version 2)' type Scripts.

Unless specified otherwise, 'Summary Page' and 'Summary Page (version 2)' will be used interchangeably throughout this document and will both refer to 'Summary Page (version 2)' type Scripts.

Styles and Formatting

Details on styling and formatting Summary Pages so that they look similar to built-in finPOWER Connect forms is given in Appendix A – Guidelines.

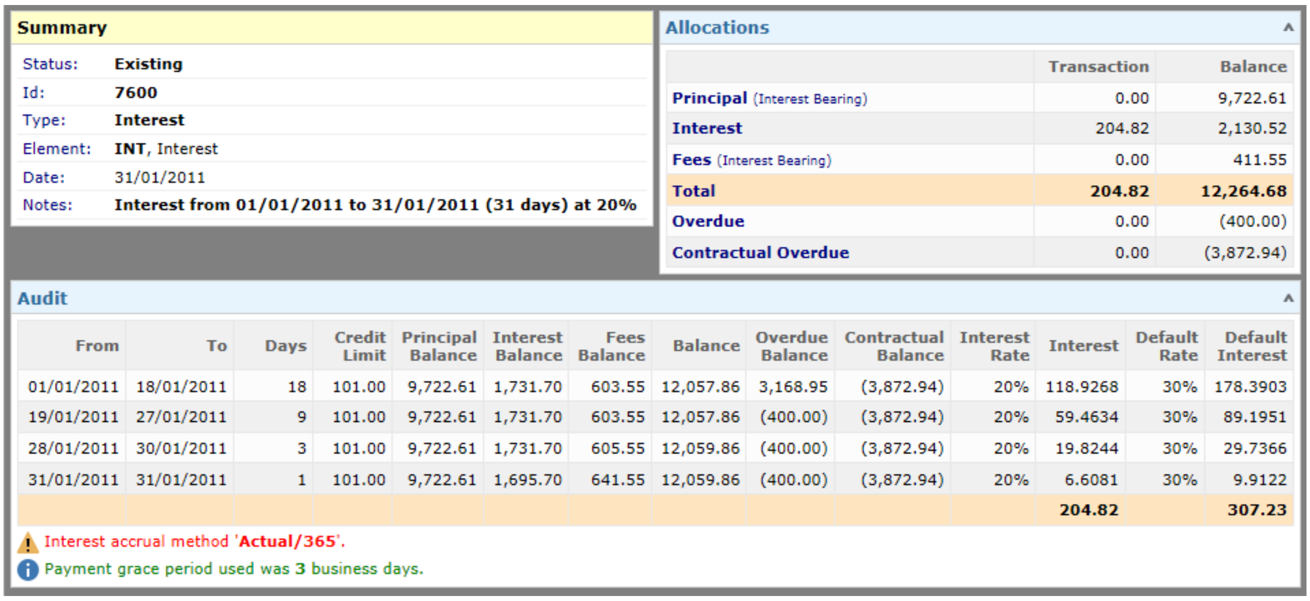
Quick References

The following quick references are contained in this document:

- Summary Table Row Classes
- Summary Table Cell Classes
- Wiki Text
- HTML Templates
- Application Shortcuts

Anatomy of a Summary Page

The following screenshot shows the layout of a Summary Page:



Summary

Status:	**Existing**
Id:	**7600**
Type:	**Interest**
Element:	**INT**, Interest
Date:	31/01/2011
Notes:	**Interest from 01/01/2011 to 31/01/2011 (31 days) at 20%**

Allocations

	Transaction	Balance
Principal (Interest Bearing)	0.00	9,722.61
Interest	204.82	2,130.52
Fees (Interest Bearing)	0.00	411.55
Total	**204.82**	**12,264.68**
Overdue	0.00	(400.00)
Contractual Overdue	0.00	(3,872.94)

Audit

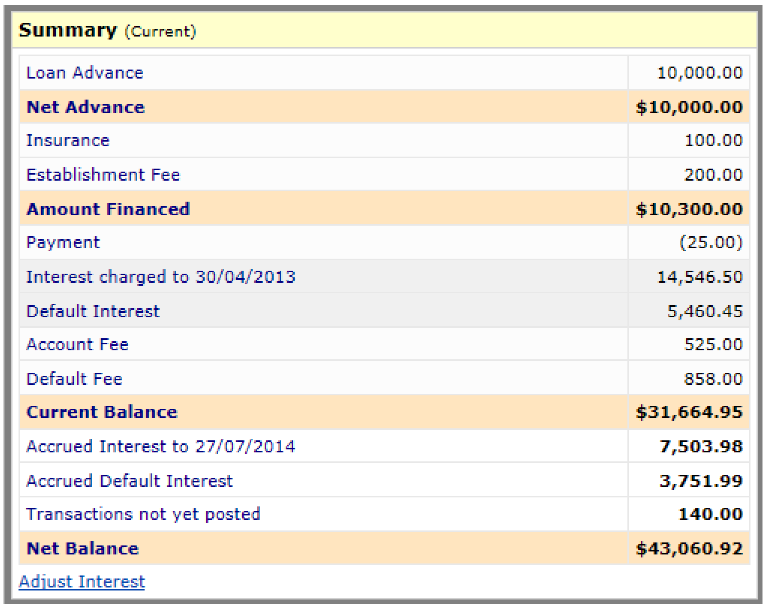
From	To	Days	Credit Limit	Principal Balance	Interest Balance	Fees Balance	Balance	Overdue Balance	Contractual Balance	Interest Rate	Interest	Default Rate	Default Interest
01/01/2011	18/01/2011	18	101.00	9,722.61	1,731.70	603.55	12,057.86	3,168.95	(3,872.94)	20%	118.9268	30%	178.3903
19/01/2011	27/01/2011	9	101.00	9,722.61	1,731.70	603.55	12,057.86	(400.00)	(3,872.94)	20%	59.4634	30%	89.1951
28/01/2011	30/01/2011	3	101.00	9,722.61	1,731.70	605.55	12,059.86	(400.00)	(3,872.94)	20%	19.8244	30%	29.7366
31/01/2011	31/01/2011	1	101.00	9,722.61	1,695.70	641.55	12,059.86	(400.00)	(3,872.94)	20%	6.6081	30%	9.9122
											204.82		**307.23**

⚠ Interest accrual method '**Actual/365**'.

ℹ Payment grace period used was **3** business days.

- This page contains two sections:
  - A two-column section containing the **Summary** and **Allocations** blocks.
    - Each column is configured to be 50% of the page width.
  - A single-column section containing the **Audit** block.
    - The column is configured to be 100% of the page width.
  - **NOTE:** Typically, a Summary Page will only contain a single, multi-column section.
- The **Summary** block:
  - Is flagged as the main block meaning:
    - It has a different colour header.
    - It is not collapsible, i.e., there is no arrow icon on the right-hand side of the heading.
  - Contains a single Summary Table with a class of 'Information'.
    - These tables generally consist of two columns; a caption and a value.
- The **Allocations** block:
  - Contains a single Summary Table with a class of 'Tabular Data'.
    - Data tables are used to display tabular data.
- The **Audit** block:
  - Contains a single Summary Table with a class of 'Tabular Data'.
  - Displays a 'Warning' and an 'Information' message after the table.

The following screenshots show variations on Summary Page blocks:

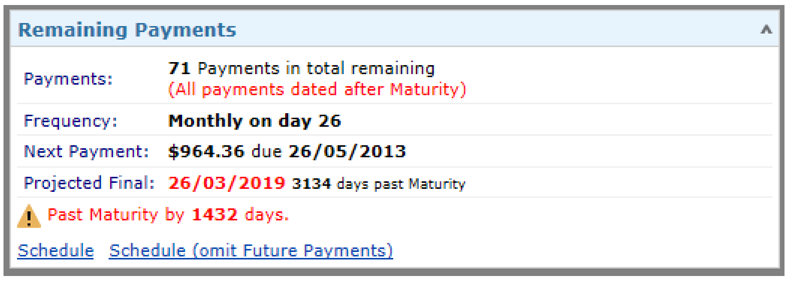


Summary (Current)

Loan Advance	10,000.00
Net Advance	**$10,000.00**
Insurance	100.00
Establishment Fee	200.00
Amount Financed	**$10,300.00**
Payment	(25.00)
Interest charged to 30/04/2013	14,546.50
Default Interest	5,460.45
Account Fee	525.00
Default Fee	858.00
Current Balance	**$31,664.95**
Accrued Interest to 27/07/2014	**7,503.98**
Accrued Default Interest	**3,751.99**
Transactions not yet posted	**140.00**
Net Balance	**$43,060.92**

Adjust Interest

- This **Summary** block:
  - Contains a small suffix of '(Current)' in its heading.
  - Shows an 'Actions' row.
    - This row contains only a single action: Adjust Interest.



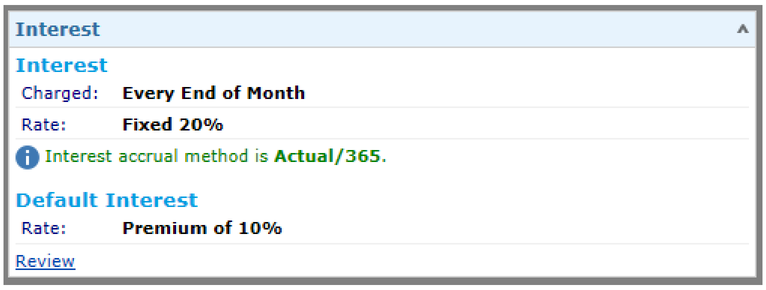
Remaining Payments

Payments:	**71** Payments in total remaining (All payments dated after Maturity)
Frequency:	**Monthly on day 26**
Next Payment:	**$964.36** due **26/05/2013**
Projected Final:	**26/03/2019** **3134** days past Maturity

Past Maturity by **1432** days.

Schedule Schedule (omit Future Payments)

- This **Remaining Payments** block:
  - Uses special 'Wiki' formatting to make some of the text values smaller, bold and red.
  - Shows a warning message, also using Wiki formatting to make '1432' appear bold.
  - Has an 'Actions' row containing two action hyperlinks.



Interest

Interest

Charged:	**Every End of Month**
Rate:	**Fixed 20%**

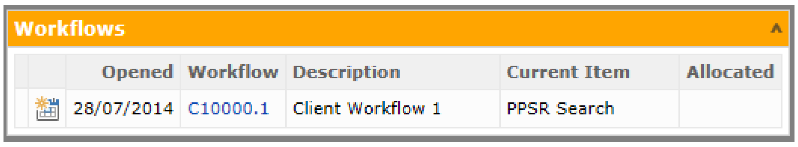
Interest accrual method is **Actual/365**.

Default Interest

Rate:	**Premium of 10%**

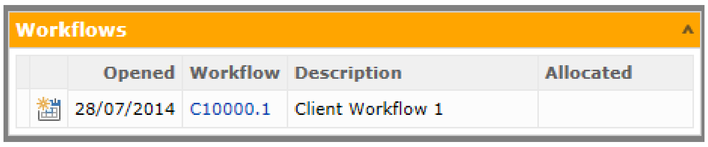
Review

- This **Interest** block:
    - Contains two summary tables:
        - Interest
            - This table has an information message.
        - Default Interest
            - This table has an 'Actions' row containing a single action hyperlink.



Workflows

	Opened	Workflow	Description	Current Item	Allocated
	28/07/2014	C10000.1	Client Workflow 1	PPSR Search	



Workflows

	Opened	Workflow	Description	Allocated
	28/07/2014	C10000.1	Client Workflow 1	

- This **Workflows** block:
    - Has a coloured heading to highlight that there are open Workflows.
    - Hides the 'Current Item' column automatically if the form is resized below a certain limit.
    - Uses an 'Icon' template to display an icon.
    - Uses an Application Shortcut to allow Workflow C10000.1 to be opened from a link.

Summary Page (version 2) Scripts

Summary Pages are generally created via 'Summary Page (version 2)' type Scripts.

This type of Script has the following function signature:

```
Public Function Main(source As Object,
                     eventId As String,
                     eventArgs As ISKeyValueList,
                     ByRef handled As Boolean,
                     ByRef returnValues As ISKeyValueList,
                     ByRef text As String) As Boolean

  ' Assume Success
  Main = True

End Function
```

The Script parameters are:

- **source**
  - The Script can be passed a 'Source' object, E.g., if the Summary Page is being displayed from the Accounts form, this will be a `finAccount` object.
  - **NOTE:** Some Scripts check to see if the source is a `String` or an `Object` and act accordingly, E.g., the built-in Account Key Details Script will create and load a `finAccount` object if the source is a `String` (the source is assumed to be the Account Id).
- **eventId**
  - Generally this will be a blank String indicating that the Script should simply return a Summary Page. Script Events are described later in this document.
- **eventArgs**
  - This is a Key/ Value list that can be used in conjunction with `eventId` to pass additional information to the Script.
- **handled**
  - The Script can set handled to `True` to indicate that it has done something. This is not necessary in most situations.
- **returnValues**
  - The Script can return (or populate) a Key/ Value list of values that can be used by a calling function.
- **text**
  - The Script should return any HTML it generates in this parameter.

ScriptInfo

The `ScriptInfo` object is available to all Scripts and contains contextual information that may be useful to the Script including:

- `Target`
  - The target for this Summary Page based on the `iseSummaryPageTarget` Enum, E.g.:
    - `Form`
      - For display on a finPOWER Connect form.
    - `Report`
      - For display in a finPOWER Connect report.
    - `WebServices`
      - For returning from finPOWER Connect Web Service.
  - This can be used to vary the output, E.g.:

        - When initialising the HTML Summary Page Builder.
            - E.g., no Action hyperlinks will be rendered when generating for a target of 'Report'.
            - Icons and styles will be embedded in the document (where possible) when generating for a target of 'WebServices'.
        - The Script can choose to vary displayed columns based on the target.
- `Constants`
    - A Key/ Value List of Script constants.
- `FormRecordMode`
    - The record mode of the form from which this Summary Page is being generated. One of the following `iseFormRecordMode` Enum `values`:
        - `Add`
            - A new record is being added.
        - `Edit`
            - An existing record is being edited.
        - `Loaded`
            - An existing record is loaded but not being edited.
        - `NoRecord`
            - No record is loaded.
    - Script code often uses this to determine whether to display action hyperlinks, E.g.:
        - The Accounts form, Status page hides actions such as reviewing Credit Limits if the record is being edited.

Summary Tables

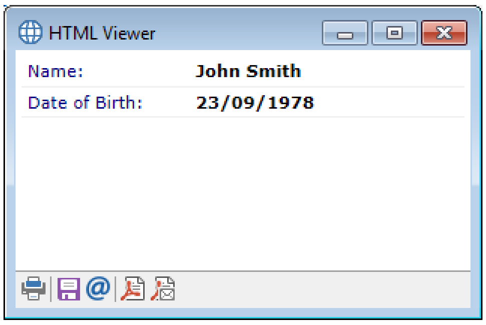
Blocks within Summary Pages almost always consist of one or more Summary Tables.

Summary Tables are represented by the `ISSummaryTable` business layer object.

The following 'Summary Page (version 2)' type Script creates a very simple Summary Table.

```
Public Function Main(source As Object,
                     eventId As String,
                     eventArgs As ISKeyValueList,
                     ByRef handled As Boolean,
                     ByRef returnValues As ISKeyValueList,
                     ByRef text As String) As Boolean

  Dim SummaryTable As ISSummaryTable

  ' Assume Success
  Main = True

  ' Create Summary Table
  SummaryTable = finBL.CreateSummaryTable()
  With SummaryTable
    .TableClass = iseSummaryTableClass.Information

    .Columns.AddText(20)
    .Columns.AddText()

    With .Rows
      .AddCaptionText("Name:", "John Smith")
      .AddCaptionDate("Date of Birth:", New Date(1978, 9, 23))
    End With
  End With

  ' Convert to HTML
  text = finBL.Utilities.GetHtmlFromSummaryTable(SummaryTable)

End Function
```

When run from the 'Test' page of the Scripts form, the following HTML is displayed:


HTML Viewer

Name:	**John Smith**
Date of Birth:	**23/09/1978**

Although this Script creates only a simple page containing a single Summary Table, variations on it will be used to demonstrate Summary Table functionality in this section.

NOTE: Although the final output is an HTML page, the Script did not define any HTML.

Summary Table Objects

Summary Tables consist of the following objects which are described in this section:

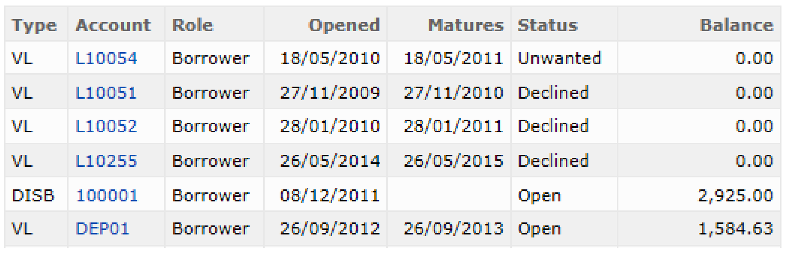
- `ISSummaryTable`
  - A single Summary Table.
- `ISSummaryTables`
  - A collection of Summary Tables.
- `ISSummaryTableColumn`
  - Details of a column in a Summary Table.
- `ISSummaryTableColumns`
  - A collection of column details for a Summary Table.
- `ISSummaryTableRow`
  - A row in a Summary Table.
- `ISSummaryTableRows`
  - A collection of rows in a Summary Table.
- `ISSummaryTableCell`
  - A cell in a Summary Table row.
- `ISSummaryTableCells`
  - A collection of cells in a Summary Table row. Typically one per column in the Summary Table.
- `ISSummaryTableAction`
  - An action to display underneath a Summary Table, typically in the form of a hyperlink.
- `ISSummaryTableAction`
  - A collection of Summary Table actions.

ISSummaryTable

Represents a single Summary Table.

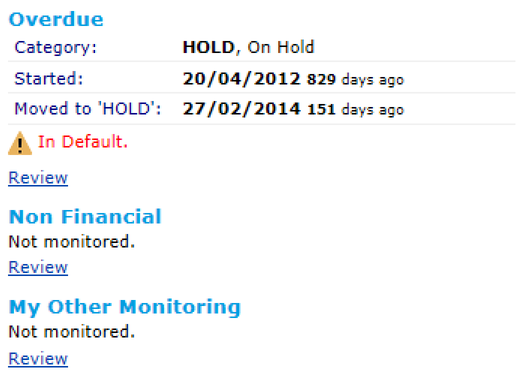
The more common Summary Table properties are:

- **AutoStripe**
  - Indicates whether the table should automatically alternate row shading (striping).
  - Typically used in 'Tabular Data' tables, E.g.:



Type	Account	Role	Opened	Matures	Status	Balance
VL	L10054	Borrower	18/05/2010	18/05/2011	Unwanted	0.00
VL	L10051	Borrower	27/11/2009	27/11/2010	Declined	0.00
VL	L10052	Borrower	28/01/2010	28/01/2011	Declined	0.00
VL	L10255	Borrower	26/05/2014	26/05/2015	Declined	0.00
DISB	100001	Borrower	08/12/2011		Open	2,925.00
VL	DEP01	Borrower	26/09/2012	26/09/2013	Open	1,584.63

- **Caption**
  - A Caption to display above the table.
  - Typically used where multiple Summary Tables are being displayed in a single block, E.g.:


Overdue

Category:	**HOLD**, On Hold
Started:	**20/04/2012 829** days ago
Moved to 'HOLD':	**27/02/2014 151** days ago

⚠ In Default.

Review

Non Financial

Not monitored.

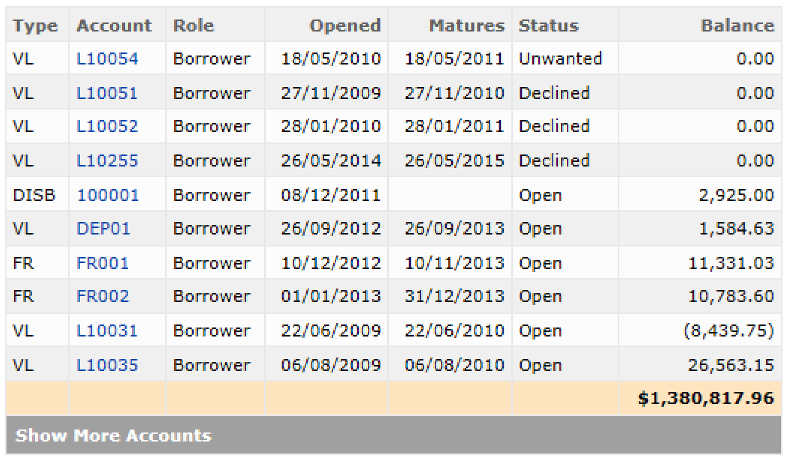
Review

My Other Monitoring

Not monitored.

Review

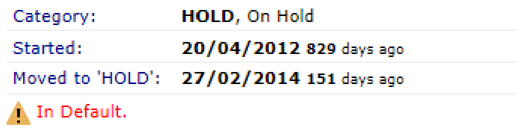
- **MaxDisplayRows**
  - Generally used for 'Tabular Data' tables, this determines the number of rows that are displayed before a 'Show Hidden Rows' link is displayed, E.g.:



Type	Account	Role	Opened	Matures	Status	Balance
VL	L10054	Borrower	18/05/2010	18/05/2011	Unwanted	0.00
VL	L10051	Borrower	27/11/2009	27/11/2010	Declined	0.00
VL	L10052	Borrower	28/01/2010	28/01/2011	Declined	0.00
VL	L10255	Borrower	26/05/2014	26/05/2015	Declined	0.00
DISB	100001	Borrower	08/12/2011		Open	2,925.00
VL	DEP01	Borrower	26/09/2012	26/09/2013	Open	1,584.63
FR	FR001	Borrower	10/12/2012	10/11/2013	Open	11,331.03
FR	FR002	Borrower	01/01/2013	31/12/2013	Open	10,783.60
VL	L10031	Borrower	22/06/2009	22/06/2010	Open	(8,439.75)
VL	L10035	Borrower	06/08/2009	06/08/2010	Open	26,563.15
						$1,380,817.96

Show More Accounts

- **ShowHiddenRowsCaption**
  - The caption to display on the link to show all rows, E.g., 'Show More Accounts'.
  - If unspecified, this will be rendered as 'Show All'.
- **SummaryTableId**
  - A unique Id to identify this Summary Table in a collection.
  - Specified when adding a Summary Table to an `ISSummaryTables` collection.
- **TableClass**
  - This defines the style of the table, either 'Information' or 'Tabular Data'.
- **TextAfter**
  - Text to display after the Summary Table.
  - This is generally one or more HTML Templates, E.g., to display a warning (in this case 'In Default'):



Category:	**HOLD**, On Hold
Started:	**20/04/2012 829** days ago
Moved to 'HOLD':	**27/02/2014 151** days ago

⚠ In Default.

Other, less commonly used properties are `DisableColSpan`, `TextBefore` and `TextRight`.

`ISSummaryTable` exposes the following collections:

- **Actions**
  - A collection of actions to display after the Summary Table.
- **Columns**

    - A collection of column details for the Summary Table.
- **Rows**
    - A collection of rows for the Summary Table.

ISSummaryTables

Represents a collection of Summary Table.

Most functions in the `finHtmlSummaryPage2StringBuilder` object accept a collection of Summary Tables.

This collection has the following methods:

- `Add(summaryTableId, summaryTable)`
    - Add an item to the end of the collection.
    - Each Summary Table in the collection must have a unique summaryTableId.
    - If only adding a single table to the collection, this will generally be given an Id or 'MAIN'.
- `Clear()`
    - Clear the collection.
- `IndexOf(summaryTableId)`
    - Returns the zero-based index of a Summary Table in the collection or -1 if a Summary Table with the specified Id is not found.
- `InsertAfter(insertAfterSummaryTableId, summaryTableId, summaryTable)`
    - Insert a Summary Table after the specified Summary Table.
    - If no Summary Table with an Id of insertAfterSummaryTableId is found, the Summary Table will be added to the end of the collection.
- `InsertBefore(insertBeforeSummaryTableId, summaryTableId, summaryTable)`
    - Insert a Summary Table before the specified Summary Table.
    - If no Summary Table with an Id of insertAfterSummaryTableId is found, the Summary Table will be added to the end of the collection.
- `Item(summaryTableId)`
    - Get a Summary Table.
    - This is the default method, therefore you can omit the word 'Item', E.g.:

      ```
      SummaryTable = SummaryTables('MAIN')
      ```
- `ItemByIndex(index)`
    - Get a Summary Table based upon its zero-based index in the collection.
- `RemoveById(summaryTableId)`
    - Remove a Summary Table from the collection.
    - If a Summary Table with the specified summaryTableId is not found, no error will occur.

ISSummaryTableColumn and ISSummaryTableColumns

Summary Tables define a collection of columns.

Generally, an 'Information' type Summary Table will have only two columns; a caption and a value.

Columns are added to the collection using one of the following methods of `ISSummaryTableColumns`. Many of these methods have overloaded function signatures, therefore their parameters are not specified in the list below.

- `AddBoolean()`
- `AddCode()`
- `AddCurrency()`
- `AddDate()`
- `AddDateTime()`
- `AddDouble()`
- `AddInteger()`
- `AddPercentage()`
- `AddText()`

NOTE: The data type of the column added, affects the default column alignment and width.

Some of the more common parameters that can be specifies with the various Add methods are:

- **Caption**
  - If specified and the Summary Table does not contain any rows (as is usual), a 'Header' row will be added to the Summary Table and a cell added to this row to display the caption.
  - **IMPORTANT:** If you wish to add a series of columns to a Summary Table and some have headers and some don't, specify a space as the caption, otherwise no cell will be added into the 'Header' row.
- **widthChars**
  - The number of characters that this column should be or -1 for unspecified.
  - **NOTE:** This is just a rough guide that is used when generating an HTML table from an `ISSummaryTable` object.

After adding a column (each of the Add methods returns an `ISSummaryTableColumn` object), the following properties can be updated:

- **CssClass**
  - Generally you will never set this but if it contains a value then, when an HTML table is generated from this Summary Table, this CSS class will be applied to all cells in this column.
- **HideRule**
  - A 'Hide Rule' can be specified so that when the HTML page hosting the Summary Table falls below a certain width, this column is hidden, E.g.:

    ```
    Column.HideRule = iseSummaryTableHideRule.DocumentWidthLessThan800px
    ```
- **MaxDisplayChars**
  - Setting this to a non-zero value will restrict how many characters a column displays.

- If the column contains more characters, an ellipsis icon will be displayed and, clicking this will display the hidden text.

- **NoBreakChars**
  - Setting this to a non-zero value will force the HTML to not line-break for this number of characters.
  - This is useful if you have a column that will generally only display short text values (e.g., you have specified the widthChars property as 10) but, if it does display a larger value, you want to ensure that no line break occurs, E.g.:

    ```
    Column.NoBreakChars = 20
    ```

- **Sortable**
  - Not currently supported.

- **TextAlign**
  - The alignment of text within the column.
  - This defaults based upon the data type of the column but can be overridden, E.g., if you have a currency column (aligns right by default) but you wish this column to display text left-aligned since it may contain some non-currency values, you could do:

    ```
    Column.TextAlign = iseSummaryTableCellTextAlign.Left
    ```

The following example adds various types of columns to a 'Tabular Data' table. This is based loosely upon the Accounts table used in the Client Key Details Summary Page:

```
' Create Summary Table
SummaryTableMain = finBL.CreateSummaryTable()
With SummaryTableMain
  .TableClass = iseSummaryTableClass.TabularData
  .AutoStripe = True
  .MaxDisplayRows = 10

  ' Columns (including Header row)
  With .Columns
    .AddCode("Type", finBL.AccountTypes.GetLongestCode())
    .AddCode("Account", finBL.AccountFunctions.GetLongestCode())
    .AddText("Description")
    .AddCode("Role", 5) ' Making a Code column means text will not word break
    .AddDate("Opened")
    .AddDate("Matures")
    .AddCode("Status", 5) ' Making a Code column means text will not word break
    .AddCurrency("Opening")
    .AddCurrency("Balance")
    .AddCurrency("Overdue")
    .AddCurrency("Contractual")
  End With
End With
```

ISSummaryTableRow and ISSummaryTableRows

Summary Tables define a collection of rows.

For a Summary Table to display any information, the table must contain at least one row.

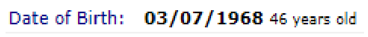
Some of the more common row properties are:

- **RowClass**
  - A row's class defines its style (see Quick Reference).
  - Typically, you would only change a row's class when dealing with 'Tabular Data' tables, E.g., to show a total row in a different style.
  - The class can be one of the following `iseSummaryTableRowClass` values:
    - `None`
    - `Normal`
    - `NormalBold`
      - Text in the row will be bold.
    - `Alt1`
      - The row background will be shaded in striping colour 1.
    - `Alt2`
      - The row background will be shaded in striping colour 2.
    - `Header`
      - Defines a header row.
    - `HeaderCollapsible`
      - No yet fully implemented.
    - `Footer`
      - Defines a footer row which will appear orange.
    - `FooterBold`
      - Defines a footer row with bold text.
    - `Negative`
      - Defines a row with a light red (pink) background.
- **Visible**
  - Can be set to `False` to hide a row.
  - If a table contains hidden rows, a special row will appear at the bottom of the table to 'Show All Rows'.

When adding rows to an 'Information' table, the following 'AddCaption' helper methods are generally used. These add a row and then append two cells to the row, one for the caption and one for the value. Many of these methods have overloaded versions of the function signature. Only the more common versions are shown:

- `AddCaptionAddressDetailsPhysical(caption, addressDetails)`
  - Adds a row containing a caption and a formatted physical address.
- `AddCaptionAddressDetailsPostal(caption, addressDetails)`
  - Adds a row containing a caption and a formatted postal address.
- `AddCaptionCodeDescription(caption, code, description)`
  - Adds a row containing a caption and a value consisting of a code and description, E.g., an Account Type Id and it's description.
  - **NOTE:** There are many variations on this method. Examples are given later in this section.

- `AddCaptionCurrency(caption, value, [includeSymbol])`
  - Adds a row containing a caption and a currency value, optionally displaying the currency symbol.
- `AddCaptionDate(caption, value, [smallSuffix])`
  - Adds a row containing a caption and a date value.
  - You can optionally specify a piece of text to display after the date (`smallSuffix`), E.g.:



Date of Birth: **03/07/1968** 46 years old

- `AddCaptionInteger(caption, value)`
  - Adds a row containing a caption and an Integer value.
- `AddCaptionPercentage(caption, value, decimalPlaces, [includeSymbol], [smallSuffix])`
  - Adds a row containing a caption and an Integer value.
  - The number of decimal places must be specified and you can optionally include the percentage symbol and a piece of text to display after the value (`smallSuffix`).
- `AddCaptionText(caption, text)`
  - Adds a row containing a caption and a text value.
- `AddCaptionTextEmailAddress(caption, emailAddress)`
  - Adds a row containing a caption and a text value representing an email.
  - A hyperlink will be created to create an email.
  - This method also has optional parameters to provide a `clientId` and `contactRecordId` (the Client Contact Method's primary key).
    - This allows the hyperlink to provide client specific options.
- `AddCaptionTextPhoneNumber(caption, phoneNumber)`
  - Adds a row containing a caption and a text value representing a phone number.
  - A hyperlink will be created to call the number.
  - This method also has optional parameters to provide a `clientId` and `contactRecordId` (the Client Contact Method's primary key).
    - This allows the hyperlink to provide client specific options, E.g., to send an SMS if licensed.
- `AddCaptionTextWebsiteUrl(caption, websiteUrl)`
  - Adds a row containing a caption and a text value representing a Website URL.
  - A hyperlink will be created to follow the URL.
  - This method also has optional parameters to provide a `clientId` and `contactRecordId` (the Client Contact Method's primary key).
    - This allows the hyperlink to provide client specific options.
- `AddCaptionTick(caption, value, [showCross])`
  - Adds a row containing a caption and a Boolean value represented as a Tick icon.
  - If value is `False`, nothing will be displayed unless the `showCross` parameter is set to `True`.
- `AddCaptionWiki(caption, text)`
  - Adds a row containing a caption and a text value formatted as simple Wiki Text.
- `AddSectionHeading1(caption)`
  - Adds a row containing a single cell (this will automatically span all columns in the rows) containing a Heading 1 style heading.

    - **NOTE:** Heading 1 style headings are rarely, if ever, used in built-in Summary Pages. Heading 2 is generally used.
- `AddSectionHeading2(caption)`
    - Adds a row containing a single cell (this will automatically span all columns in the rows) containing a Heading 2 style heading.

Typically, when building a Summary Table, you would add all table rows in order however, if you wish to modify an existing Summary Table, it may be desirable to be able to insert rows in between other rows and delete certain rows. The following methods cater for this:

- `InsertAfter(insertAfterCaption)`
    - This returns an `ISSummaryTableRow` object inserted into the rows collection after a row with the specified caption (i.e., the first cell in the row's Cells collection matches this caption).
        - If `insertAfterCaption` is not found, the new row will be added to the end of the collection.
- `InsertBefore(insertBeforeCaption)`
    - This returns an `ISSummaryTableRow` object inserted into the rows collection before a row with the specified caption (i.e., the first cell in the row's Cells collection matches this caption).
    - If `insertBeforeCaption` is not found, the new row will be added to the end of the collection.
- `RemoveByCaption(caption)`
    - Removes a row based upon the specified caption (i.e., the first cell in the row's Cells collection matches this caption).

More information of updating existing Summary Tables is given in the Overriding Built-In Blocks section.

ISSummaryTableCell and ISSummaryTableCells

Summary Tables rows define a collection of cells.

The number of cells in a row would typically match the number of columns defined for the Summary Table however, this is not necessary since if too few cells are specified for a row, the last cell will span all remaining columns.

The methods in the previous section, E.g., `ISSummaryTableRows.AddCaptionText()` simply created a new row and added two cells to the row. This is the typical method used when creating 'Information' tables. However, when creating 'Tabular Data' tables, it is typical to first add a row and then append cells to that row, E.g.:

```
' Create Summary Table
SummaryTableMain = finBL.CreateSummaryTable()
With SummaryTableMain
  .TableClass = iseSummaryTableClass.TabularData
  .AutoStripe = True

  ' Columns
  With .Columns
    .AddText(" ", 1, iseSummaryTableCellTextAlign.Center) ' Flag Colour
    .AddDate("Opened")
    .AddCode("Workflow", finBL.WorkflowFunctions.GetLongestCode())
    .AddText("Description").HideRule = iseSummaryTableColumnHideRule.DocumentWidthLessThan640px
    .AddCode("Allocated", finBL.Users.GetLongestCode())
  End With

  ' Rows
  With .Rows
    For Each AccountWorkflow In account.Workflows
      ' Add Row
      With .Add().Cells
        .AddText("{{flag|" & AccountWorkflow.FlagColour & "}}")
        .AddDate(AccountWorkflow.OpenedDate)
        .AddText(AccountWorkflow.WorkflowId)
        .AddText(AccountWorkflow.Description)
        .AddText(AccountWorkflow.AllocatedUserId)
      End With
    Next
  End With
End With
```

Some of the more common cell properties are:

- **CellClass**
  - A cell's class defines its style (see Quick Reference).
  - The class can be one of the following `iseSummaryTableCellClass` values:
    - `None`
    - `Normal`
    - `NormalBold`
    - `Caption`
    - `CaptionBold`
    - `SectionHeading1`
      - **NOTE:** Built-in Summary Pages rarely, if ever, use this style. `SectionHeading2` is generally used.
    - `SectionHeading2`
    - `Highlight1`
    - `Highlight2`
    - `Positive`
    - `PositiveBold`
    - `Small`
    - `SmallBold`

        - `Warning`
        - `WarningBold`
- **ColSpan**
    - When set to a value greater than one, the cell will span more than one column.
- **NoWrap**
    - Force the cell to never wrap text.
- **TextAlign**
    - The cell's text alignment.
    - If left at `Default`, this will be inherited from the table column's value which is determined by the data type when adding a column.
- **Value**
    - The cell's value as a String.
- **ValueFormat**
    - The cell's value can be one of the following `iseSummaryTableCellFormat` values:
        - `Text`
            - Plain text.
        - `WikiTextSimple`
            - Simple, Wiki formatted text.

When adding cells to a row, the following 'Add' helper methods are generally used. Many of these methods have overloaded versions of the function signature. Only the more common versions are shown:

- `AddAddressDetailsPhysical(addressDetails)`
    - Adds a cell containing a formatted physical address.
- `AddAddressDetailsPostal(addressDetails)`
    - Adds a cell containing a formatted postal address.
- `AddCaption(value, [textAlign], [bold])`
    - Adds a cell containing a text caption.
    - The cell is given a `CellClass` of either `Caption` or `CaptionBold`.
- `AddCaptionWiki(value, [textAlign], [bold])`
    - Adds a cell containing a Wiki formatted text caption.
    - The cell is given a `CellClass` of either `Caption` or `CaptionBold` and a `ValueFormat` of `WikiTextSimple`.
- `AddCurrency(value, [includeSymbol], [blankIfZero], [textAlign])`
    - Adds a cell containing a currency value.
- `AddDate(value, [longDateFormat], [textAlign])`
    - Adds a cell containing a date value.
- `AddDateTime(value, [longDateFormat], [longTimeFormat], [textAlign])`
    - Adds a cell containing a date/ time value.
- `AddDouble(value, decimalPlaces, [blankIfZero], [textAlign])`
    - Adds a cell containing a Double value.
- `AddInteger(value, [blankIfZero], [textAlign])`
    - Adds a cell containing an Integer value.
- `AddPercentage(value, decimalPlaces, [blankIfZero], [textAlign])`

    - Adds a cell containing a Double value formatted with a percentage symbol.
- `AddText(value)`
    - Adds a cell containing a text value.
- `AddTick(value, [showCross], [center])`
    - Adds a cell containing a Boolean value displayed as a Tick.
    - If value is `False` then nothing will be displayed unless `showCross` is `True`.
    - The icon can optionally be centered in the cell by setting `center` to `True`.
- `AddWiki(value)`
    - Adds a cell containing a Wiki formatted text value.

ISSummaryTableAction and ISSummaryTableActions

Summary Tables rows define a collection of actions.

Actions are displayed as hyperlinks after the table.

Actions are generally defined by Application Shortcuts and, in the case of built-in Summary Pages, generally excute a Form Action. The only exception to this is the special 'custom' type actions available on certain forms.

Each action in the collection is given a unique Action Id when adding the the collection, E.g.:

```
' Create Summary Table
SummaryTableMain = finBL.CreateSummaryTable()
With SummaryTableMain
  .TableClass = iseSummaryTableClass.Information

  ' Actions
  With .Actions
    .AddFormAction("CreditLimitReview", "Review")
    .AddFormAction("PendingWithdrawalReview", "Review Pending Withdrawal")
  End With
End With
```

Some of the more common action properties are:

- **ActionId**
  - The action's unique identifier.
- **ApplicationShortcutType**
  - This can be any valid Application Shortcut Type, E.g., FormAction, FormShow etc.
  - A special 'FormActionCustom' type is supported.
    - This renders the Application Shortcut with a 'custom://' prefix rather than the normal 'app://' prefix and is required by custom actions on certain forms.
- **Available**
  - A Boolean value indicating wether the action is available.
    - Unavailable actions will not be rendered.
    - An Action may be made unavailable due to User Permission or other reasons.
  - **NOTE:** Simply not adding the action to the collection if it is not available achieves the same result.
- **Caption**
  - The display caption for the action.
- **Parameters**
  - A Key/ Value List of Application Shortcut parameters.
  - **NOTE:** May of the methods used to add actions to the collection allow the parameters to specified as a URL String for simplicity.

The following methods are used to add Actions to the collection:

- `AddApplicationShortcut(actionId, caption, applicationShortcut, [available])`
  - Add an Application Shortcut action.
- `AddApplicationShortcutUrl(actionId, caption, url, [available])`
  - Add an Application Shortcut action.
- `AddFormAction(actionId, caption, [available], [parametersUrl])`
  - Add a 'FormAction' type Application Shortcut action.
- `AddFormCustomAction(actionId, caption, [available], [parametersUrl])`
  - Add a special custom action.

    - This will be rendered with a special 'custom://' prefix and is required by custom action on certain forms.
- `AddFormShow(actionId, caption, form, [available], [parametersUrl])`
    - Add a 'FormShow' type Application Shortcut action.
- `AddHtmlShowFunction(actionId, caption, functionName, [title], [parametersUrl], [available])`
    - Add an 'HtmlShowFunction' type Application Shortcut action.

Information Tables

Typically, 'Information' tables display two columns; a caption and a value.

Some tables, E.g., the Summary table on the Accounts form, Status page break this rule and display three columns. Others such as the 'Also Known As' table in the Client Key details summary section use only one column.

Generally, rows are added to an Information table using a single function call, E.g., `AddCaptionText()`, E.g.:

```
With SummaryTable
  .TableClass = iseSummaryTableClass.Information

  .Columns.AddText(20)
  .Columns.AddText()

  With .Rows
    .AddCaptionText("Name:", Client.Name)
    .AddCaptionDate("Date of Birth:", Client.DateOfBirth)
  End With
End With
```

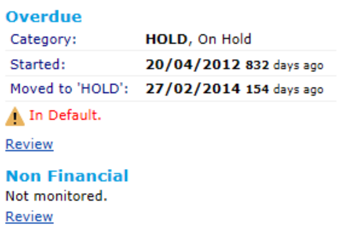
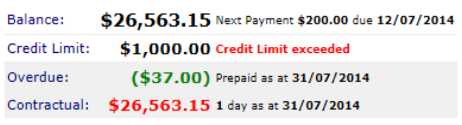
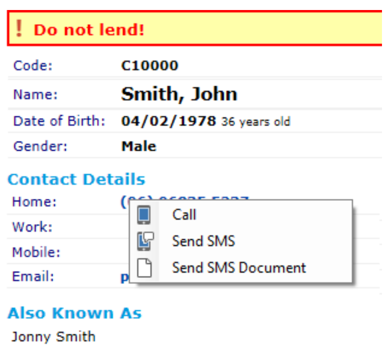
Built-In Examples

Viewing the built-in code to generate Information type Summary Tables is the best way to learn how to achieve a particular layout or format.

The easiest way to view built-in code is to do the following:

- Open the Scripts form.
- Click the 'Add' button.
- Change the 'Type' to 'Summary Page Standard Block Override'.
- On the Script Code page, right-click in the Script editor.
  - Select **Insert Standard Block (Expanded)**
  - Select a standard block to insert, E.g., Account, Account_Monitoring.

The table below lists a selection of standard blocks and what they demonstrate:

Standard Block	Demonstrates
`Account_Monitoring`  **Overdue** Category: **HOLD**, On Hold Started: **20/04/2012 832** days ago Moved to 'HOLD': **27/02/2014 154** days ago ⚠ In Default. Review **Non Financial** Not monitored. Review	Returning multiple Summary Tables (one for each Monitor Category).
	Setting the table's caption which then appears as a heading above the table.
	Adding a Code/ Description based on a global collection.
	Using TextAfter to show HTML templates, E.g., a Warning.
	Actions after each table.
`Account_FinancialStatus`  Balance: **$26,563.15** Next Payment **$200.00** due **12/07/2014** Credit Limit: **$1,000.00** **Credit Limit exceeded** Overdue: **($37.00)** Prepaid as at **31/07/2014** Contractual: **$26,563.15** **1** day as at **31/07/2014**	Using three columns in an Information table.
	Using Wiki text.
	Highting cell text by making it larger (the Balance).
	Highlighting rows by changing their background colour (Overdue rows are light grey).
`Client_Summary`  **! Do not lend!** Code: **C10000** Name: **Smith, John** Date of Birth: **04/02/1978** 36 years old Gender: **Male** **Contact Details** Home: (Work: Mobile: Email: p **Also Known As** Jonny Smith	Using the 'Alert' HTML template.
	Using hyperlinks in cells, E.g.: • Open the Clients form (not when viewed from Clients form). • In Wiki text (for 'Special Type' Clients). • Show a popup menu of phone number actions.
	A single column Information table, i.e., 'Also Known As'.
	Wiki text after a date (to show age).

Tabular Data Tables

Typically, 'Tabular Data' tables display a grid of information.

Some tables, E.g., the Summary table on the Accounts form, Financial page use only two columns so may easily be mistaken for 'Information' type tables.

Unlike Information tables, where you typically add a row with two cells using a single function call, E.g., `AddCaptionText()`, when generating a Tabular Data table you add a row and then add cells to that row, E.g.:

```
With SummaryTableClients
  .TableClass = iseSummaryTableClass.TabularData
  .Caption = "Clients"

  ' Columns
  With .Columns
    .AddCode("Code", 10)
    .AddText("Name")
  End With

  For Each SecurityStmtClient In SecurityStmt.Clients
    ' Add Row
    With .Rows.Add()
      .Cells.AddText(SecurityStmtClient.ClientId)
      .Cells.AddText(SecurityStmtClient.ClientName)
    End With
  Next
End With
```

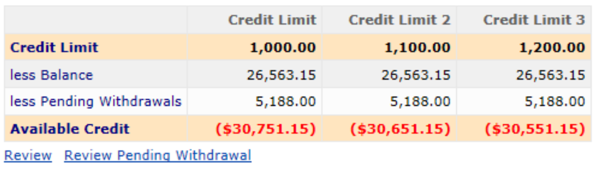
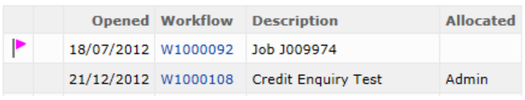
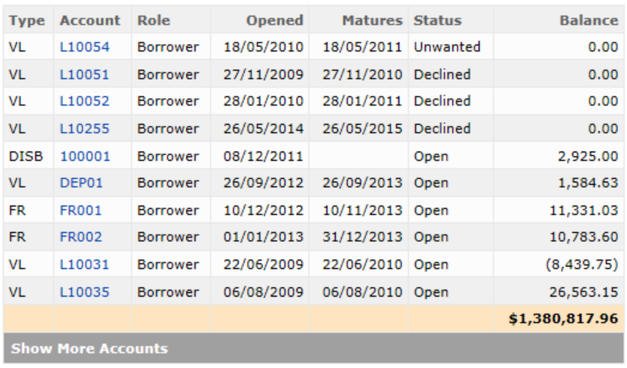
Built-In Examples

Viewing the built-in code to generate Tabular Data type Summary Tables is the best way to learn how to achieve a particular layout or format.

The easiest way to view built-in code is to do the following:

- Open the Scripts form.
- Click the 'Add' button.
- Change the 'Type' to 'Summary Page Standard Block Override'.
- On the Script Code page, right-click in the Script editor.
    - Select **Insert Standard Block (Expanded)**
    - Select a standard block to insert, E.g., Account, Account_CreditLimits

The table below lists a selection of standard blocks and what they demonstrate:

Standard Block	Demonstrates
`Account_CreditLimits` 	Varying the number of columns.
	Varying row styles, E.g., footer style rows.
	Setting the foreground colour of cells to 'red' if negative.
	Actions after the table.
`Account_IncompleteWorkflows` 	Using the Icon and Flag HTML templates.
	Cell hyperlinks to open the Workflows form.
	Responsive columns that hide if the page is narrow.
`Client_Accounts2` 	Displays a maximum of 10 rows and provides a link to 'Show More Accounts'.
	Hides rows (by setting `Visible = False`) for closed Accounts.
	Total row.
	Auto-striping of table rows.
	Varies column widths based on longest codes in database, E.g., Account column's width.

	Credit Limit	Credit Limit 2	Credit Limit 3
Credit Limit	**1,000.00**	**1,100.00**	**1,200.00**
less Balance	26,563.15	26,563.15	26,563.15
less Pending Withdrawals	5,188.00	5,188.00	5,188.00
Available Credit	**($30,751.15)**	**($30,651.15)**	**($30,551.15)**

Review Review Pending Withdrawal

	Opened	Workflow	Description	Allocated
⚑	18/07/2012	W1000092	Job J009974	
	21/12/2012	W1000108	Credit Enquiry Test	Admin

Type	Account	Role	Opened	Matures	Status	Balance
VL	L10054	Borrower	18/05/2010	18/05/2011	Unwanted	0.00
VL	L10051	Borrower	27/11/2009	27/11/2010	Declined	0.00
VL	L10052	Borrower	28/01/2010	28/01/2011	Declined	0.00
VL	L10255	Borrower	26/05/2014	26/05/2015	Declined	0.00
DISB	100001	Borrower	08/12/2011		Open	2,925.00
VL	DEP01	Borrower	26/09/2012	26/09/2013	Open	1,584.63
FR	FR001	Borrower	10/12/2012	10/11/2013	Open	11,331.03
FR	FR002	Borrower	01/01/2013	31/12/2013	Open	10,783.60
VL	L10031	Borrower	22/06/2009	22/06/2010	Open	(8,439.75)
VL	L10035	Borrower	06/08/2009	06/08/2010	Open	26,563.15
						$1,380,817.96

Show More Accounts

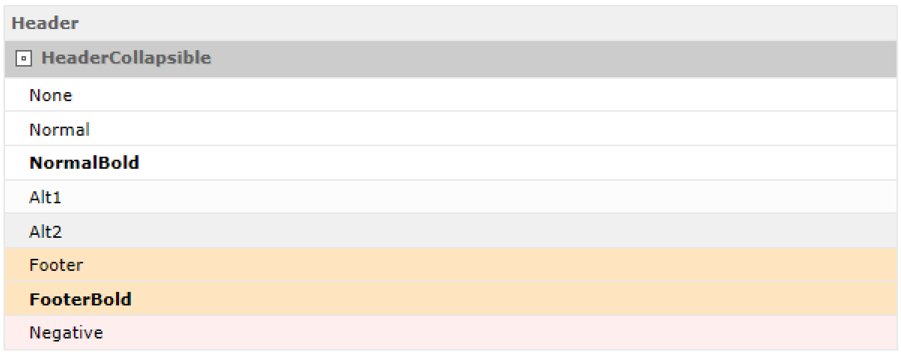
Quick Reference

Row Classes

The following table lists all available row classes (defined by the `iseSummaryTableRowClass` Enum) and their usage:

Class	Description
`Header`	Header row.
`HeaderCollapsible`	Collapsible header row (not yet fully implemented).
`None`	Undefined style.
`Normal`	Normal style.
`NormalBold`	Bold.
`Alt1`	Alt background colour 1.
`Alt2`	Alt background colour 1.
`Footer`	Footer row.
`FooterBold`	Bold footer row.
`Negative`	Highlighted row (for negative reasons).

This screenshot shows how the various row classes are rendered in HTML:



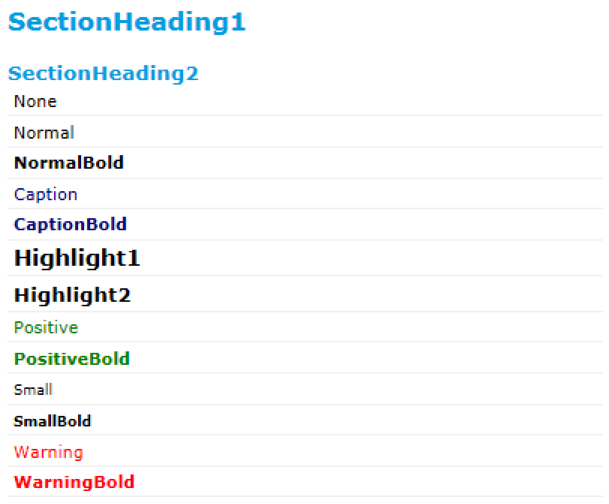
Header
HeaderCollapsible
None
Normal
NormalBold
Alt1
Alt2
Footer
FooterBold
Negative

Cell Classes

The following table lists all available row classes (defined by the `iseSummaryTableCellClass` Enum) and their usage:

Class	Description
`SectionHeading1`	Section Heading 1 (not generally used).
`SectionHeading2`	Section Heading 2.
`None`	Undefined style.
`Normal`	Normal style.
`NormalBold`	Bold.
`Caption`	Caption, E.g., the caption cell in an Information table.
`CaptionBold`	Bold caption.
`Highlight1`	Highlight figure 1 (Information tables only).
`Highlight2`	Highlight figure 2 (Information tables only).
`Positive`	Positive value or message.
`PositiveBold`	Bold positive value or message.
`Small`	Small text.
`SmallBold`	Small, bold text.
`Warning`	Warning value or message.
`WarningBold`	Bold warning value or message.

This screenshot shows how the various cell classes are rendered in HTML:


SectionHeading1
SectionHeading2
None
Normal
NormalBold
Caption
CaptionBold
Highlight1
Highlight2
Positive
PositiveBold
Small
SmallBold
Warning
WarningBold

NOTE: Wiki text is used for further text formatting within a cell.

Advanced Formatting

Hiding Columns

Hiding columns based on the width of the Summary Page generally applies only to Tabular Data tables.

Columns are hidden by setting their `HideRule` property to one of the following `iseSummaryTableColumnHideRule` Enum properties:

- `TargetHtml`
  - Hide if the Summary Table is being used to generate an HTML Summary Page.
  - **NOTE:** Currently, Summary Tables can only be used for HTML Summary Pages so this value exists for future use.
- `TargetReport`
  - Hide if the Summary Page Builder is targeting a report.
- `DocumentWidthLessThan240px`
  - Hide if HTML document is less than 240 pixels wide.
- `DocumentWidthLessThan320px`
  - Hide if HTML document is less than 320 pixels wide.
- `DocumentWidthLessThan400px`
  - Hide if HTML document is less than 400 pixels wide.
- `DocumentWidthLessThan500px`
  - Hide if HTML document is less than 500 pixels wide.
- `DocumentWidthLessThan640px`
  - Hide if HTML document is less than 640 pixels wide.
- `DocumentWidthLessThan800px`
  - Hide if HTML document is less than 800 pixels wide.

Use hiding of columns sparingly and ensure only 'less important' columns are hidden.

Hiding Rows

Hiding (including restricting the number of rows visible by default) applies only to Tabular Data tables.

Rows can be hidden in one of two ways:

- By setting the `MaxDisplayRows` property of the Summary Table.
  - E.g., setting this to 10 will, by default, show only 10 rows and provide a link to 'Show All Rows'.
    - The link caption can be changed using the `ShowHiddenRowsCaption` property.
- By setting the `Visible` property of a row to `False`.
  - If a table has any hidden rows, a 'Show All Rows' link will be added to the bottom of the table.

Depending on the `ScriptInfo.Target` property, hiding rows may have no effect. For example if the target is `iseSummaryPageTarget.Report`.

Preventing Text from Line Breaking

Preventing text within a Summary Table from breaking over multiple lines occurs automatically in the following circumstances:

- The following Column types will never break text over multiple lines:
    - Code
    - Date
    - DateTime
- A Column's `NoBreakChars` property has been set to a non-zero value.
    - E.g., setting this to 10 will prevent any cell text from breaking for at least the first 10 characters, regardless of how narrow the column is, E.g.:

      ```
      column breaks
      here
      ```

      NOTE: HTML typically only breaks at certain characters, hence the above example does not line-break in the middle of the word 'breaks'.
- A Cell's `NoWrap` property is set to `True`.

HTML Summary Page Builder

A `finHtmlSummaryPage2StringBuilder` object is used to create version 2 Summary Pages.

The process for creating a Summary Page generally follows these steps:

- Create the Summary Page Builder.
- Initialise the Summary Page Builder.
- Create a section and add columns and blocks.
- Finalise the Summary Page Builder and resolve any HTML Templates.

The following is a highly stripped down version of the built-in Client Summary page:

```
Private Function CreatePage(ByRef html As String) As Boolean

  Dim Ok As Boolean
  Dim sb As finHtmlSummaryPage2StringBuilder

  ' Assume Success
  Ok = True

  ' Initialise
  sb = finBL.CreateHtmlSummaryPage2StringBuilder()

  ' Initialise HTML
  With sb
    .Initialise(ScriptInfo)
    .HtmlHeaderBegin("", isefinSummaryPageStyle.Blocks)
    .HtmlHeaderEnd()
  End With

  ' Begin Section
  sb.SectionColumnsBegin()

  ' Begin Column 1
  sb.ColumnBegin("60%")

  ' Summary
  sb.BlockBegin("Summary", "", , False)
  sb.SummaryTablesAppend(finBL.SummaryPageStandardBlocks.Client_Summary(ScriptInfo, mClient)
  sb.BlockEnd()

  ' Accounts
  sb.BlockBegin("Accounts", "", isefinSummaryPageBlockHeadingStyle.Normal, True)
  sb.SummaryTablesAppend(finBL.SummaryPageStandardBlocks.Client_Accounts2(ScriptInfo, mClient,
ShowAccountsClosed, False, ShowAccountDescription, ShowAccountOpening, ShowAccountOverdue,
ShowAccountContractualOverdue, ShowAccountTotals, ShowAccountsMax, Nothing))
  sb.BlockEnd()

  ' End Column 1
  sb.ColumnEnd()

  ' Begin Column 2
  sb.ColumnBegin("40%", "25em", "50em")

  ' Addresses
  sb.BlockBegin("Addresses")
  sb.SummaryTablesAppend(finBL.SummaryPageStandardBlocks.Client_Addresses(ScriptInfo, mClient))
  sb.BlockEnd()

  ' End Column 2
  sb.ColumnEnd()

  ' End Section
  sb.SectionColumnsEnd()

  ' Finalise HTML
  sb.HtmlFinalise()

  ' Resolve Templates
  html = sb.ToString(True)

  Return Ok

End Function
```

NOTE: No HTML is used directly and each block is populated by calling one of the built-in standard block functions, all of which simply return an ISSummaryTables collection.

Initialising

Before using the Summary Page Builder, it must first be initialised.

This configures the builder, e.g., so it knows whether the HTML is to target a Form, Report or Web Services and can therefore vary its output.

It also creates HTML headers (depending on the target).

```
' Initialise
sb = finBL.CreateHtmlSummaryPage2StringBuilder()

' Initialise HTML
With sb
  .Initialise(ScriptInfo)
  .HtmlHeaderBegin("Document Title", isefinSummaryPageStyle.Blocks)
  .HtmlHeaderEnd()
End With
```

The following methods are used in the initialisation process:

- `Initialise(ScriptInfo)`
  - Initialises the builder.
  - By passing in `ScriptInfo`, the Target (Form, Report, Web Services etc) can be used to vary output, E.g.:
    - Exclude actions when rendering Summary Tables for Reports.
    - Embed styles and icons (where possible) when targeting Web Services.\
  - **NOTE:** A different version of the `Initialise` method allows the Summary Page Target and other information to be passed in rather than `ScriptInfo`.
- `HtmlHeaderBegin([title], [style], [compact])`
  - Creates HTML headers (where required).
  - If `title` is specified, the HTML will include a `<title>` tag.
  - The `style` can be one of the following `isefinSummaryPageStyle` Enum values:
    - `None`
    - `Blocks`
      - This is the normal style for Summary Pages and provides a dark grey background for the page.
    - `Document`
      - This style is not commonly used and is currently only used for some internal finPOWER pages.
  - If `compact` is `True` then a special CSS class will be applied to the document that reduces spacing between certain HTML elements meaning the document can fit into a smaller area.
    - The 'Quick Search' results shown in the finPOWER Connect Taskpane use the compact style.
- `HtmlHeaderEnd()`
  - Closes HTML headers where required.

Summary Page Target

When initialising the Summary Page Builder, the 'Target' is determined either from the passed in `ScriptInfo` object or is passed directly into the `Initialise` method.

The Target is an `iseSummaryPageTarget` value and can be one of the following:

- `Form`
  - The Summary Page is being produced for display within a finPOWER Connect Windows form.
- `Report`
  - The Summary Page is being produced for display within a finPOWER Connect report.
- `WebInterface`
  - The Summary Page is being produced for display within the finPOWER Connect Web Interface.
  - **NOTE:** The Web Interface is obsolete, i.e., no further Summary Page changes will be made for this target.
- `WebServices`
  - The Summary Page is being produced to return from a Web Services call.
- `WebServicesPartial`
  - The Summary Page is being produced to return from a Web Services call.
  - **NOTE:** The 'Partial' part means that the caller has requested a Summary Page that can be included in another HTML document, e.g., it does not contain any <html> or <body> tags.
- `WebExternal1`
  - The Summary Page is being produced to return from an external Web application.
- `WebExternal2`
  - The Summary Page is being produced to return from an external Web application.
- `External1`
  - The Summary Page is being produced to return from an external application.

Form

When producing HTML for this target, the following take place:

- Any Summary Table Columns with a 'Hide Rule' of '' are not rendered.
  - **NOTE:** This functionality is not currently implemented.

Report

When producing HTML for this target the following take place:

- Block collapsers are not shown since there is no concept of collapsing and expanding blocks in reports.
- Hidden rows in Summary Tables are ignored; rows are always visible.
- Hyperlinks in Summary Table cells are not rendered.
- Text is never truncated in Summary Table cells.
- Summary Table Cell tooltips are ignored.
- 'Hide Rules' for Summary Table columns are not applied.
- Summary Table Actions are not rendered.

Web Services

When producing HTML for this target the following take place:

- The Summary Page 2 CSS stylesheet is included inline in the generated HTML.
- Images such as icons are embedded within the HTML where possible, e.g., where the image is generated through an HTML template such as {{icon}}.

Web Services Partial

When producing HTML for this target the following take place:

- Instead of producing a full HTML document, the Summary Page content is enclosed in a <div> block with all relevant CSS classes applied, e.g.:
    - <div class='is-summarypage-2 is-compact is-summarypage-blocks'>
      content
      </div>

Sections

As shown in the Anatomy of a Summary Page section, a Summary Page generally consists of a single section but can contain more if necessary, E.g., you want to vary the number of columns in each section.

A section is divided up into one or more columns and these columns can be defined as a percentage of the overall document width or a fixed size.

The `SectionColumnsBegin()` and `SectionColumnsEnd()` methods are used to start and end a section and, within the section, the `ColumnBegin()` and `ColumnEnd()` methods are used to start and end columns, E.g.:

```
' Begin Section
sb.SectionColumnsBegin()

' Begin Column 1
sb.ColumnBegin("60%")

' Insert Blocks Here

' End Column 1
sb.ColumnEnd()

' Begin Column 2
sb.ColumnBegin("40%", "25em", "50em")

' Insert Blocks Here

' End Column 2
sb.ColumnEnd()

' End Section
sb.SectionColumnsEnd()
```

The above sample adds a section containing two columns. The first column is configured to be 60% of the document width and the second column, 40% of the document width with a minimum width of 25ems and a maximum of 50ems.

NOTE: In HTML, ems are a unit of measurement that is relative to the font-size being used, E.g., if the font size is 12pt then 25ems will produce a wider column than if the font size is 10pt.

The `ColumnBegin()` method has the following parameters:

- **width**
  - An HTML width for the column, E.g.:
    - 60%
    - 200px
    - 40em
  - **NOTE:** Typically you would use a percentage, e.g., 60% and 40% or 50% and 50% for a two column section.
- **minWidth**
  - A HTML width specifying the minimum width that this column should be.
  - This prevents the column being too narrow if the User resizes the form (e.g., the Clients form) to a small size.
- **maxWidth**
  - A HTML width specifying the maximum width that this column should be.
  - This prevents the column from becoming too wide if the form (e.g., the Clients form) is maximised on a high resolution screen.

NOTE: A Summary Table can also define overriding widths for when rendering as HTML, e.g.:

```
SummaryTable.MaxWidthHtml = "40em"
SummaryTable.WidthHtml = "30em"
SummaryTable.WidthHtml = "auto"
```

Where `"auto"` is a special value that tells the HTML Summary Page renderer to not apply the specified table width, i.e., just let HTML size it normally.

Blocks

Blocks are added within columns on the Summary Page. Each block typically displays a different type of information, e.g.:

'TODO: Screenshot.

A block typically displays one or more Summary Tables although here are some exceptions such as the 'Map' block in the built-in Client Summary page that inserts HTML directly in order to embed Google Maps.

Wiki Text

In the absence of using HTML directly, Summary Pages can use a simple form of mark-up to produce formatted text.

Wiki Text allows basic formatting such as bold, small text and hyperlinks.

The following table shows all available Wiki Text instructions supported by Summary Pages:

Formatting	Wiki	Example
Bold	`**`	`This is **Bold**` This is **Bold**
Small	`--`	`This is --Small--` This is Small
Warning	`!!`	`This is !!a Warning!!` This is a Warning
Positive	`++`	`This is ++Positive++` This is Positive
Link	`[[url]]` or `[[url|caption]]`	`Intersoft's site [[http://www.intersoft.co.nz]] and [[http://www.google.com|Google]].` Intersoft's site http://www.intersoft.co.nz and Google.
New Line	`<br>` or `vbLineLine`	`Line 1<br>Line 2<br>Line 3` Line 1 Line 2 Line 3

Wiki instructions can be combined, e.g.:

```
This Account is !!Overdue!! by **45** --days--.
**[[app://FormAction?action=ShowSchedule&index=1|Schedule]]**.
```

This Account is Overdue by **45** days.

View the **Schedule**.

Summary Tables

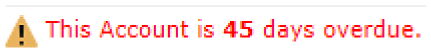
To use Wiki Text in a Summary Table, the cell displaying the Wiki text must have a `ValueFormat` of `iseSummaryTableCellFormat.WikiTextSimple`.

The cell's format call be set directly but usually, one the following methods would be used:

- `ISSummaryTableCells.AddWiki(value)`
  - Adds a cell containing a Wiki formatted text.
- `ISSummaryTableRows.AddCaptionWiki(caption, text)`
  - Add a row containing two cells with the second cell configured to display Wiki text.

In addition to Summary Table cells containing Wiki text, many HTML templates used by the table (generally specified by the Summary Table's `TextAfter` property) also use Wiki text, e.g.:

```
SummaryTableMain.TextAfter = "{{warning|This Account is **45** days overdue.}}"
```


This Account is **45** days overdue.

HTML Templates

Calling the templates used by 'Summary Page (version 2)' type Scripts 'HTML Templates' is a bit of a misnomer; most templates that support formatted text actually use Wiki text rather than HTML.

However, since the original version 1 Summary Page Scripts referred to these as HTML templates, this naming convention has been retained.

HTML Templates are used to provide advanced, consistent formatting within Summary Pages, e.g., to create a warning message or display an icon.

HTML Templates are always defined within double curly braces, e.g.:

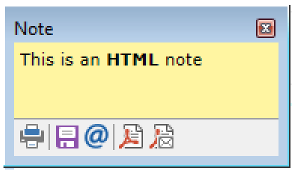
```
{{Warning|This is a **warning** message}}
```

This is a **warning** message

Many, simple templates like the warning message above use the pipe character to separate the template name (e.g., 'warning') from the template content (e.g., 'this is a warning message').

Other, more complex, templates use named parameters, e.g.:

```
{{PopupNote note='This is an <b>HTML</b> note' icon='Note' tooltip='click to view note'}}
```

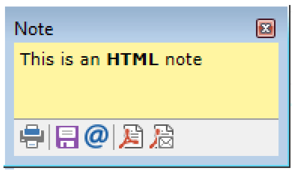

Note

This is an **HTML** note

WARNING: Do not use the `String.Format()` method when creating HTML Templates from Script code since the curly braces will conflict with the way this .NET method works.

Quick Reference

The following tables lists all common HTML Templates available to Summary Page Scripts:

Template	Description	Example
Alert	Inserts an alert message.	`{{Alert|This Client should not be loaned to.}}` ! This Client should not be loaned to.
Button	Inserts a button that will not show when the page is printed.	`{{Button caption='Show It' icon='Account' hyperlink='app://formshow?form=Accounts'}}` Show It
ClientImage	Inserts a Client's image.	`{{ClientImage clientId='C10000' width='100'}}`
ColourBlock	Inserts a square, coloured block.	`{{ColourBlock|red}} {{ColourBlock|#00FF00}}` 🟥 🟩
Cross	Inserts a cross icon.	`{{Cross}}` ✗
Error	Inserts an error message.	`{{Error|Failed to load Client 'C10000'.}}` ! Failed to load Client 'C10000'.
Flag	Inserts a coloured flag.	`{{Flag|Orange}} {{Flag|#0000FF|Blue Flag}}` ⚑ ⚑
Icon	Inserts an icon.	`{{Icon|Account}} {{Icon|Client}}`
Info	Inserts an information message.	`{{Info|This Account is due to mature in **7** days.}}` i This Account is due to mature in **7** days.
PopupNote	Inserts an icon that launches a popup HTML note.	`{{PopupNote note='This is an <b>HTML</b> note'}}`  Note This is an **HTML** note
ResourceImage	Inserts a Resource Image.	`{{ResourceImage resourceId='PIGGY' width='200'}}`

Tick	Inserts a tick icon.	`{{Tick}}` ✓
Warning	Inserts a warning message.	`{{Warning|This Account is Overdue by **45** days.}}` ⚠ This Account is Overdue by **45** days.

Action

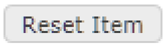
Inserts an application shortcut (or other) hyperlink but styles it the same as the 'Actions' that appear below Summary Tables.

This template has the following named parameters:

- **caption**
  - The button caption.
- **hyperlink**
  - The hyperlink to follow when the hyperlink is clicked.

Examples of this template are:

```
{{Action caption='Reset Item' hyperlink='custom://resetitem&index=1'}}
```



WARNING: Do not confuse this template with the 'Button' template.

Address

Inserts an address including a lookup hyperlink.

This template has the following named parameters:

- **addressDetailsXml**
  - The button caption.
- **singleLine**
  - A Boolean value indicating whether to format the address as a single line.
  - **NOTE:** Enclose value in single quotes since XML formatting uses double quotes.
- **physicalFormat**
  - A Boolean value indicating whether to format the address according to physical address rules rather than postal address rules.

As an alternative to `addressDetailsXml`, the following address parts can be specified.

- **instructions**
- **streetAddress** (or streetAddressFull)
- **suburb**
- **city**
- **state**
- **postcode**
- **country**

Examples of this template are:

```
{{Address addressDetailsXml='<ISAddressDetails>…'}}

{{Address streetAddress='1 Oak Road' city='Napier' postcode='4000'}}
```

Alert

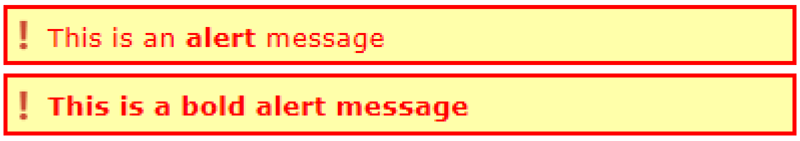
Inserts an alert message.

This template has the following parameters, separated by a pipe character:

- **message**
  - A Wiki formatted message.
- **bold**
  - A Boolean value indicating whether to make the message bold.

Examples of this template are:

```
{{Alert|This is an **alert** message}}

{{Alert|This is a bold alert message|True}}
```



! This is an **alert** message

! **This is a bold alert message**

Button

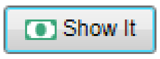
Inserts a button that will not show when the page is printed.

This template has the following named parameters:

- **bold**
  - A Boolean value indicating whether to make the caption bold.
- **caption**
  - The button caption.
- **captionColour**
  - The caption colour or blank for the default colour.
- **hyperlink**
  - The hyperlink to follow when the button is clicked.
- **icon**
  - The button's icon resource id or blank to not show an icon.
- **iconSize**
  - The button's icon size or blank for the default 16 pixel icon.
- **tooltip**
  - The tooltip text to display.

Examples of this template are:

```
{{Button caption='Show It' icon='Account' hyperlink='app://formshow?form=Accounts'}}
```



CheckBox

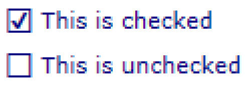
Inserts a CheckBox image.

This template has the following parameters, separated by a pipe character:

- **value**
    - A Boolean value.
- **caption**
    - The Caption to display.

Examples of this template are:

```
{{CheckBox|True|This is checked }}

{{CheckBox|False|This is unchecked }}
```



☑ This is checked

☐ This is unchecked

ClientImage

Inserts a Client's image.

This template has the following, named, parameters:

- **clientId**
  - The Id of the finPOWER Connect Client.
- **width**
  - The image width.
  - If omitted, a width of 100 pixels is assumed.

Examples of this template are:

```
{{ClientImage clientId='C10000'}}

{{ClientImage clientId='C10000' width='50'}}
```

WARNING: If the Document Manager is unavailable, no image will be inserted.

ColourBlock

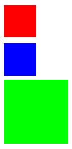
Inserts a colour block.

This template has the following parameters, separated by a pipe character:

- **colour**
  - An HTML colour.
- **tooltip**
  - A tooltip to display when hovering over the colour block.
- **width**
  - The colour block's width. Not generally specified.
  - If omitted, a width of 16 pixels is assumed.
- **height**
  - The colour block's height. Not generally specified.
  - If omitted, a height of 16 pixels is assumed.

Examples of this template are:

```
{{ColourBlock|Red}}

{{ColourBlock|Blue|This is a 'Blue' block}}

{{ColourBlock|#00FF00|This is a 'Green' block|32|32}}
```



Cross

Inserts a cross icon.

Examples of this template are:

```
{{Cross}}
```

✗

Disclaimer

Inserts a disclaimer message.

This template has the following parameters, separated by a pipe character:

- **message**
    - A Wiki formatted message.

Examples of this template are:

```
{{Disclaimer|This is an **disclaimer** message}}
```

This is a **disclaimer** message.

NOTE: Typically, this template will only be used on 'Credit Report' type Summary Pages.

DocumentManagerImage

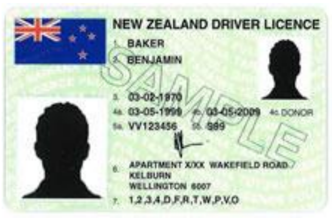
Inserts an image stored in the Document Manager.

This template has the following, named, parameters:

- **fileName**
  - A file name relative to the Document Manager folder in the format itemType\itemId\fileName, e.g.:
    - Clients\C10000\DriversLicence.png
  - **NOTE:** If the full file name is specified, the Document Manager path will first be removed and the file name re-formed to ensure the file is in the Document Manager path.
- **width**
  - The image width in HTML units, e.g.:
    - 100%
    - 500px
- **maxWidth**
  - The maximum width to display the image in HTML units, e.g.:
    - 600px

Examples of this template are:

```
{{DocumentManagerImage fileName='Clients\C10000\DriversLicense.png'}}

{{DocumentManagerImage fileName='Clients\C10000\DriversLicense.png' maxWidth='600px'}}

{{DocumentManagerImage fileName='Clients\C10000\DriversLicense.png' width='100%'
maxWidth='200px'}}
```



NEW ZEALAND DRIVER LICENCE
1. BAKER
2. BENJAMIN
SAMPLE
3. 03-02-1970
4a. 03-05-1999 4b. 03-05-2009 4c. DONOR
5a. VV123456 5b. 999
6. APARTMENT X/XX WAKEFIELD ROAD
KELBURN
WELLINGTON 6007
7. 1,2,3,4,D,F,R,T,W,P,V,O

If the file is not found in the Document Manager folder, the file name will be displayed as follows:

DriverLicencex.jpg

WARNING: If the Document Manager is unavailable, no image will be inserted.

Error

Inserts an error message.

This template has the following parameters, separated by a pipe character:

- **message**
  - A Wiki formatted message.
- **bold**
  - A Boolean value indicating whether to make the message bold.

Examples of this template are:

```
{{Error|This is an **error** message}}

{{Error|This is a bold error message|True}}
```

This is an **error** message

This is a bold error message

Flag

Inserts a coloured flag icon.

This template has the following parameters, separated by a pipe character:

- **colour**
  - An HTML colour.
  - If this is blank then no flag will be inserted.
- **tooltip**
  - A tooltip to display when hovering over the flag.

Examples of this template are:

```
{{Flag|Red}}

{{Flag|Blue|A 'Blue' Flag}}
```

Help

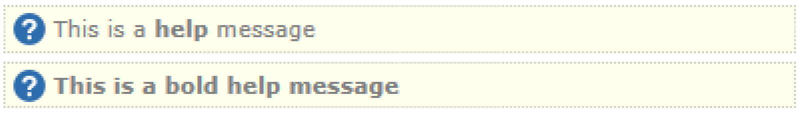
Inserts a help message.

This template has the following parameters, separated by a pipe character:

- **message**
  - A Wiki formatted message.
- **bold**
  - A Boolean value indicating whether to make the message bold.

Examples of this template are:

```
{{Help|This is a **help** message}}

{{Help|This is a bold help message|True}}
```



? This is a **help** message

? **This is a bold help message**

Icon

Inserts an icon.

This template has the following parameters, separated by a pipe character:

- **icon**
  - A valid finPOWER Connect icon name.
  - **NOTE:** A list of icons can be viewed from the 'Icon' dropdown on the General page of either the Page Sets or Workflow Types forms.
  - If the value contains a comma, the text after the comma will be assumed to be the name of an overlay icon, e.g.:
    - `Account,Add`
    - `Client,Search`
- **tooltip**
  - A tooltip to display when hovering over the icon.
- **size**
  - The icon's size. Not generally specified.
  - If omitted, a size of 16 pixels is assumed.

Examples of this template are:

```
{{Icon|Account}}

{{Icon|Client|This is Client 'C10000'}}

{{Icon|Client|This is Client 'C10000'|32}}

{{Icon|Client,Warning|This is Client 'C10000'|32}}
```

Info

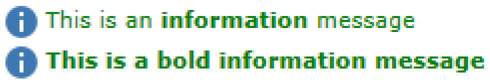
Inserts an information message.

This template has the following parameters, separated by a pipe character:

- **message**
  - A Wiki formatted message.
- **bold**
  - A Boolean value indicating whether to make the message bold.

Examples of this template are:

```
{{Info|This is an **information** message}}

{{Info|This is a bold information message|True}}
```



This is an **information** message

This is a bold information message

PopupNote

Inserts a popup note icon which, when clicked, shows a popup window displaying an HTML formatted note.

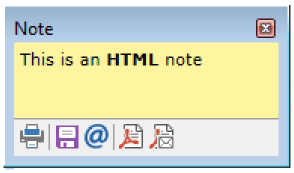
This template is only designed for displaying small, HTML snippets. See the Script Events section for details on creating more advanced popups.

This template has the following, named, parameters:

- **note**
  - An HTML formatted message.
- **icon**
  - A valid finPOWER Connect icon name or omit to use the default 'Note' icon.
  - **NOTE:** A list of icons can be viewed from the 'Icon' dropdown on the General page of either the Page Sets or Workflow Types forms.
- **tooltip**
  - A tooltip to display when hovering over the icon.
  - If omitted, this will be 'Click to view notes'.
- **title**
  - The title to display on the popup form used to display the HTML.

Examples of this template are:

```
{{PopupNote note='This is an <b>HTML</b> note'}}

{{PopupNote note='This is the note' icon='Account' tooltip='Click to view Account notes'
title='Account Notes'}}
```


Note

This is an **HTML** note

NOTE: This is one of the few places within Summary Pages where HTML can be directly specified.

ResourceImage

Inserts a Resource image.

This template has the following, named, parameters:

- **resourceId**
  - The Id of the finPOWER Connect image-type Resource.
- **width**
  - The image width.
  - If omitted, a width of 200 pixels is assumed.

Examples of this template are:

```
{{ResourceImage resourceId='PIGGY'}}

{{ResourceImage clientId='PIGGY' width='100'}}
```

SystemColour

Inserts a system colour as an HTML formatted colour String, e.g., #F0F0F0.

This template has the following parameters, separated by a pipe character:

- **systemColourName**
  - The system colour name which can be one of the following:
    - Window
      - The Window background colour used in finPOWER Connect. This varies depending on the selected theme.

Examples of this template are:

```
<body style='background-color:{{systemColour|Window}}'>
```

This template can be used in a borderless HTML Panel within a Page Set to make the HTML content blend in to the Page Set form.

Tick

Inserts a tick icon.

Examples of this template are:

```
{{Tick}}
```

✓

Warning

Inserts a warning message.

This template has the following parameters, separated by a pipe character:

- **message**
  - A Wiki formatted message.
- **bold**
  - A Boolean value indicating whether to make the message bold.

Examples of this template are:

```
{{Warning|This is a **warning** message}}

{{Warning|This is a bold warning message|True}}
```

⚠ This is a **warning** message

⚠ **This is a bold warning message**

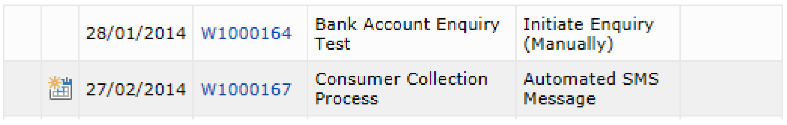
Application Shortcuts

Application Shortcuts are used to perform various tasks within finPOWER Connect.

Application Shortcuts are usually represented by a URL, e.g.:

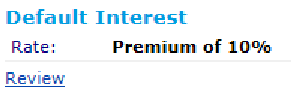
```
app://FormShow?form=Accounts&id=L10000
```

Application Shortcuts are used regularly in hyperlinks in Summary Pages, e.g., to open a form and display a display a record:



		28/01/2014	W1000164	Bank Account Enquiry Test	Initiate Enquiry (Manually)	
		27/02/2014	W1000167	Consumer Collection Process	Automated SMS Message	

Or to perform a special, form-based action:



Default Interest

Rate:	**Premium of 10%**

Review

The **app://** prefix is only used when an Application Shortcut is being run from a URL within an HTML page; it simply tells the Summary Page to interpret the link as an Application Shortcut.

NOTE: Many types of Application Shortcut detailed in this section may not be relevant to Summary Pages but all types are discussed, making this section the central reference for Application Shortcuts within finPOWER Connect.

Creating an Application Shortcut

Application Shortcuts can be created either using the finPOWER business layer or by manually building a URL String.

Using the Business Layer

The `ISApplicationShortcut` object is used to represent an Application Shortcut.

Although longer-winded than manually building a URL String, business layer Application Shortcuts have the following advantages:

- Parameters are automatically URL encoded.
  - Date parameters are automatically formatted correctly for inclusion in a URL.
- Code may be more readable.
- Parameters can be set to business layer objects.
  - E.g., you could pass a `finAccount` object to another Summary Page to view Account details.
  - **NOTE:** Passing of business layer objects is not supported from URL-based Application Shortcuts, i.e., you cannot provide a hyperlink on a Summary Page that references an object.

The following example creates an Application Shortcut:

```
Dim ApplicationShortcut As ISApplicationShortcut

' Create Application Shortcut
ApplicationShortcut = finBL.CreateApplicationShortcut()
With ApplicationShortcut
  .Action = "FormShow"

  ' Parameters
  With .Parameters
    .SetString("form", "Accounts")
    .SetString("id", "L10035")
    .SetString("page", "Logs*")
  End With
End With
```

A URL String can then be extracted for inclusion in a Summary Page, e.g.:

```
Url = ApplicationShortcut.ToUrlString(True)
```

Or, if the Summary Page is using a Script Event, this could be executed immediately as follows:

```
finBL.ExecuteApplicationShortcut(ApplicationShortcut)
```

Helper methods are available to create the various types of Application Shortcut as at finPOWER Connect version 2.02.00. Examples of these are given for each type of Application Shortcut in the next sections but some of the more common methods are:

- `finBL.CreateApplicationShortcutExternal()`
- `finBL.CreateApplicationShortcutFormAction()`
- `finBL.CreateApplicationShortcutFormShow()`

Helper methods can be used to create an ISApplicationShortcut object or to create a URL directly, e.g.:

```
' Create Application Shortcut
ApplicationShortcut = finBL.CreateApplicationShortcutFormShow("Accounts", "L10035", "Logs*")
```

```
' Create URL
Url = finBL.CreateApplicationShortcutFormShow("Accounts", "L10035", "Logs*").ToUrlString(True)
```

Building a URL String

Building a URL String directly is more suited to simple Application Shortcuts such as opening a form to show a particular record, e.g.:

```
Url = String.Format("app://FormShow?form=Clients&id={0}",
finBL.Runtime.HtmlUtilities.UrlEncode(Client.ClientId))
```

NOTE: As Per any URL, any parameters must be URL-encoded to avoid a conflict with special characters such as '&' appearing in a parameter value. When creating a URL String directly, this can be achieved using the `finBL.Runtime.HtmlUtilities.UrlEncode()` method.

When formatting dates for inclusion in an Application Shortcut URL, the **M/d/yyyy** format should be used, e.g., **8/21/2014** (8th August 2014).

Summary Tables

Application Shortcuts can be used in Summary Tables in the following ways:

- Actions (from the Summary Table's `Actions` collection).
- As a link specified in the Wiki text of an HTML Template.
- As a link within a table cell.

Actions

'TODO:

Wiki Text

'TODO:

Cell Link

'TODO:

Quick Reference

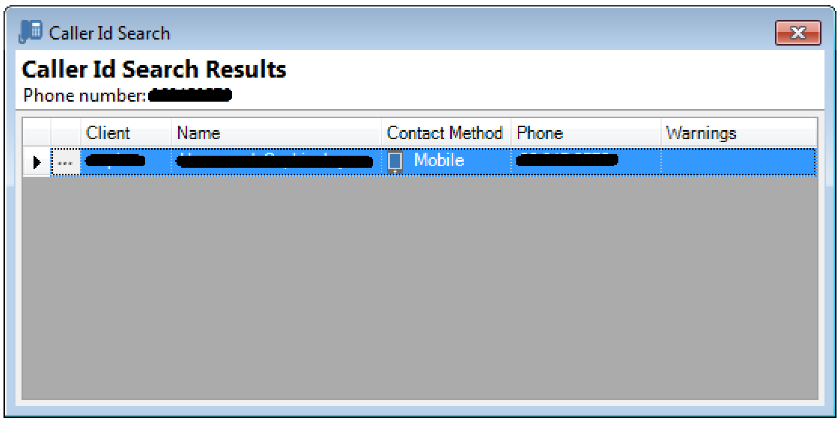
The following tables lists all, non-deprecated, Application Shortcut types (actions) apart from Workspace which basically just chains several other Application Shortcuts together:

Action	Description	Example
CallerId	Show the Caller Id Search form.	`CallerId?phoneNumber=8355237`
Contact	Show a popup menu of contact actions.	`Contact?clientId=C10000&emailAddress=js@intersoft.zzz`
Document	Run a finPOWER Connect Document.	`Document?id=AA&accountIds=L10000`
External	Open external document.	`External?file=C:\Advice.pdf`
FormAction	Perform an action on a form.	`FormAction?action=SendSMS`
FormShow	Show a form.	`FormShow?form=Accounts&id=L10000`
Global	Run a global Application Shortcut.	
HelpTopic	Show a help topic.	`HelpTopic?id=Queue.General`
HtmlShow	Show HTML or URL.	`HtmlShow?html=My HTML`
HtmlShowFunction	Show HTML from built-in function.	`HtmlShowFunction?function=AccountWarningsSummary&accountId=L10000`
HtmlShowScriptEvent	Call a Script Event to return HTML and show.	`HtmlShowScriptEvent?scriptId=SP.TEST&eventId=ShowTransactions&accountId=L10000`
MapShow	Show a map for an address.	`MapShow?address=19 Marine Parade, Napier, NZ`
MenuCommand	Call a menu command.	`MenuCommand?id=Tools.SettingsUser`
PageSet	Run a Page Set.	`PageSet?id=LoanApp`
PhoneDial	Dial a phone number.	`PhoneDial?phoneNumber=06835523712`
Queue	Run a Queue.	`Queue?id=EomReports&autoRun=True`
ScheduledScript	Run a scheduled Script.	`ScheduledScript?id=TestImport&autoRun=True&intervalMinutes=10`
Script	Run a Script.	`Script?id=TestImport`
ScriptEvent	Run a Script event.	`ScriptEvent?id=SP.TEST&eventId=StartCollectionWorkflow&accountId=L10000`

CallerId

Show the Caller Id Search form.

When passed a phone number, this Application Shortcut will show the special 'Caller Id Search' form, e.g.:



Caller Id Search Results

Phone number: [redacted]

	Client	Name	Contact Method	Phone	Warnings
...	[redacted]	[redacted]	Mobile	[redacted]	

This form searches for all Clients where a match with the supplied phone number is found.

This Application Shortcut has the following parameters:

- **phoneNumber**
  - A full or partial phone number.
- **autoShow**
  - A Boolean value indicating whether automatically show the Clients for if only one match is found.

Examples of this Application Shortcut are:

```
CallerId?phoneNumber=8355237
```

This type of Application Shortcut would typically be used by an external application via the finPOWER Connect Message Handler.

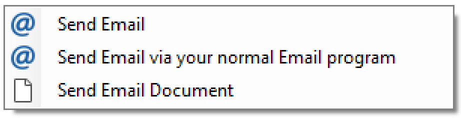
As of finPOWER Connect 2.02.00, the following helper method can be used to create an Application Shortcut:

```
ApplicationShortcut = finBL.CreateApplicationShortcutCallerId("8355237")
```

Contact

Show a popup menu of contact actions.

The options shown in the popup menu will vary depending on the parameters supplied and, if applicable, the specified Client, e.g.:



This Application Shortcut has the following parameters:

- **clientId**
  - The Id of a Client.
  - Can be left blank if any of the other parameters are specified.
- **clientContactMethodPk**
  - The primary key of a Client Contact Method record.
  - If specified, this will be used to taylor the popup menu, e.g., an option to send an SMS message will only be included if the Contact Method can receive SMS messages.
- **emailAddress**
  - The email address to contact.
  - Not necessary if `clientContactMethodPk` is specified.
- **phoneNumber**
  - The phone number to contact.
  - Not necessary if `clientContactMethodPk` is specified.
- **websiteUrl**
  - The Website URL to open.
  - Not necessary if `clientContactMethodPk` is specified.
- **phoneSms**
  - A Boolean value (default `True`) indicating whether `phoneNumber` supports SMS messages.
  - Not necessary if `clientContactMethodPk` is specified.
- **phoneVoice**
  - A Boolean (default `True`) value indicating whether `phoneNumber` supports voice calls.
  - Not necessary if `clientContactMethodPk` is specified.

Examples of this Application Shortcut are:

```
Contact?clientId=C10000&clientContactMethodPk=514

Contact?clientId=C10000&emailAddress=js@intersoft.zzz

Contact?phoneNumber=068355237&phoneSms=False
```

As of finPOWER Connect 2.02.00, the following helper methods can be used to create an Application Shortcut:

```
ApplicationShortcut = finBL.CreateApplicationShortcutContact(Client.ClientId,
Client.ContactMethods.GetCurrentEmail().Pk)

ApplicationShortcut = finBL.CreateApplicationShortcutContactEmail("C10000", "js@intersoft.zzz")

ApplicationShortcut = finBL.CreateApplicationShortcutContactPhone("", "068355237", False, True)
ApplicationShortcut = finBL.CreateApplicationShortcutContactWebsite("",
"http://www.intersoft.co.nz")
```

CreditEnquiry

Show the Credit Enquiry form to perform a follow-up Credit Enquiry.

WARNING: This type of Application Shortcut is for system use only.

This Application Shortcut has the following parameters:

- **serviceLogPk**
  - The primary key of Service Log created by the original Credit Enquiry.
- **key**
  - A service-specific key that is used to identify the type of follow-up Credit Enquiry.

Examples of this Application Shortcut are:

```
CreditEnquiry?serviceLogPk=1234&key=PossibleMatches.0
```

NOTE: Currently this type of Application Shortcut is only implemented to perform a follow-up Credit Enquiry.

The `FormShow` Application Shortcut can be used to show the Credit Enquiry form for a particular Client or Applicant, e.g.:

```
FormShow?form=CreditEnquiry&ClientId=C10000
```

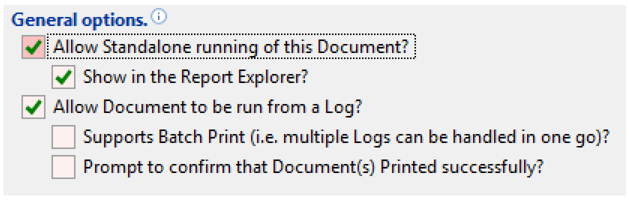
```
FormShow?form=CreditEnquiry&AccountAppId=AA10000&AccountAppApplicantPk=123
```

Document

Run a finPOWER Connect Document.

This is designed to run Documents that are configured for standalone running (rather than just from a Log such as an Account Log). Such documents generally specify parameters, e.g., a range of Accounts to include and are run from the Report Explorer, Documents group.

The ability to be run standalone is configured on the Options page of the Documents form, e.g.:



General options.

- [✓] Allow Standalone running of this Document?
  - [✓] Show in the Report Explorer?
- [✓] Allow Document to be run from a Log?
  - [ ] Supports Batch Print (i.e. multiple Logs can be handled in one go)?
  - [ ] Prompt to confirm that Document(s) Printed successfully?

The Document is run without any User Interface being displayed, e.g., when running an Account Document, the Account Documents wizard will not be displayed. Because of this, the User does not have the ability to enter parameters. Therefore, if the Document requires parameters, these must be specified as part of the Application Shortcut.

Use a FormShow Application Shortcut to display the Account Documents wizard if you wish to give the User the ability to enter parameters. The Account Documents wizard can receive special parameters to default the Document.

This Application Shortcut has the following parameters:

- **id**
  - The Id of the Document.

NOTE: Any other parameters specified as part of the Application Shortcut will be used to update the Document's parameters.

Examples of this Application Shortcut are:

```
Document?id=AA&accounts=L10000,RC10000
```

As of finPOWER Connect 2.02.00, the following helper method can be used to create an Application Shortcut:

```
Dim Parameters As ISKeyValueList

Parameters = finBL.CreateKeyValueList()
Parameters.SetString("Accounts", "L10000,RC10000")

ApplicationShortcut = finBL.CreateApplicationShortcutDocument("AA", Parameters)
```

External

Open an external document.

This uses the built-in Windows functionality to open a document based upon it's file extension. For example a file with a .PDF extension is likely to be opened in a PDF viewer such as Adobe Reader.

This Application Shortcut has the following parameters:

- **file**
  - The full file name.
- **bookmark**
  - If specified for an MS Word document (.doc, .docx), the document will be loaded and Word will scroll to the bookmark.
  - If specified for an MS Excel workbook (.xls, .xlsx), the workbook will be loaded and Excel will jump to a named range.

Examples of this Application Shortcut are:

```
External?file=C:\Test.pdf

External?file=C:\Test.doc&bookmark=MyBookmark
```

As of finPOWER Connect 2.02.00, the following helper method can be used to create an Application Shortcut:

```
ApplicationShortcut = finBL.CreateApplicationShortcutExternal("c:\Docs\Advice.doc")
```

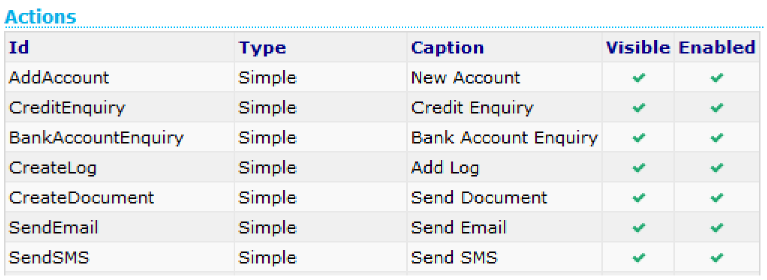
FormAction

Execute a form action.

Most forms in finPOWER Connect define special 'Actions'. To view a list of actions for a form:

- Open the form.
- Right-click form tab and select **Special**, **Form Details**.
- Locate the 'Actions' section of the 'Form Details' summary.

The following shows a subset of actions supported by the Clients form:



Actions

Id	Type	Caption	Visible	Enabled
AddAccount	Simple	New Account	✓	✓
CreditEnquiry	Simple	Credit Enquiry	✓	✓
BankAccountEnquiry	Simple	Bank Account Enquiry	✓	✓
CreateLog	Simple	Add Log	✓	✓
CreateDocument	Simple	Send Document	✓	✓
SendEmail	Simple	Send Email	✓	✓
SendSMS	Simple	Send SMS	✓	✓

Any one of these actions, providing it is enabled, can be called from an Application Shortcut.

The enabled state of an action may depend on many factors, e.g.:

- Licenced Add-Ons.
  - E.g., SMS actions will not be available unless licensed for the SMS Add-On.
- User permissions.
- The state of the form.
  - E.g., certain actions may not be available if the form is in 'Edit' mode.
- The record type
  - E.g., a 'Revolving Credit Loan' Account may support different actions to a 'Fixed Term Loan' Account.
- Configuration settings.
  - E.g., SMS actions will not be available if the User does not have an SMS provider configured.

NOTE: See Action Scripts for details on executing a Form Action defined by an Action type Script.

This Application Shortcut has the following parameters:

- **form**
  - The form key.
  - This is available from the 'Form Details' summary described above.
  - If omitted, the action will be directed at the current form.
  - A special value of '*' will direct this action to ALL forms.
- **action**
  - The Id of the Action to execute.
  - This is available from the 'Form Details' summary described above.
  - **NOTE:** Prior to finPOWER Connect version 2.02.00, this is case-sensitive.
- **open**

    - A Boolean value indicating whether to open the form if it is not already open.
    - E.g., you may wish to open the Clients form and execute the CreditEnquiry action.
- **activate**
    - A Boolean value indicating whether to activate the specified form if it is not the current form.

NOTE: Any other parameters specified as part of the Application Shortcut will be sent to the relevant built-in Form Action, e.g., the example below shows that the 'ShowSchedule' action receives an 'index' parameter.

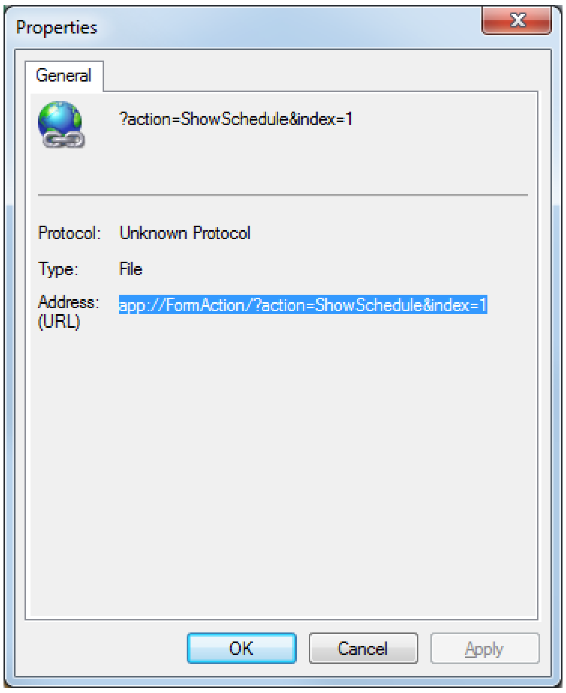
See Action Scripts for details on executing a Form Action defined by an Action type Script.

Examples of this Application Shortcut are:

```
FormAction?action=SendEmail

FormAction?form=*&action=RecordClear

FormAction?action=ShowSchedule&index=1
```

Parameters accepted by actions are not fully documented. Therefore, it may be necessary to look at built-in Summary Pages to see which parameters they use.

Alternatively, if a built-in Summary Page has a hyperlink to execute an action, right-click it and select **Properties**. This will show the URL including any additional parameters, e.g.:



?action=ShowSchedule&index=1

Protocol: Unknown Protocol

Type: File

Address: (URL) app://FormAction/?action=ShowSchedule&index=1

NOTE: Ignore the way that Internet Explorer has added a slash after the text 'FormAction'. This should not be included when creating Application Shortcut URLs.

As of finPOWER Connect 2.02.00, the following helper method can be used to create an Application Shortcut:

```
Dim Parameters As ISKeyValueList

Parameters = finBL.CreateKeyValueList()
```

```
Parameters.SetInteger("Index", 1)

ApplicationShortcut = finBL.CreateApplicationShortcutFormAction("", "ShowSchedule", Parameters)
```

FormShow

Show a form and optionally load a record and go to the specified page. The state and position of a form can also be updated from this type of Application Shortcut.

Most forms in finPOWER Connect can be opened from this type of Application Shortcut. Any form that can have it's form tab dragged to the Task Pane supports being opened from an Application Shortcut.

To show a form, you must know the form's Form Key. To view the Form Key:

- Open the form.
- Right-click form tab and select **Special**, **Form Details**.
- The Form Key is shown in the 'General' section at the top of the 'Form Details' summary.

This Application Shortcut has the following common parameters:

- **form**
  - The form key.
  - This is available from the 'Form Details' summary described above.
- **id**
  - The Id of the record to display, e.g., a Client Id.
  - For records without a String Id, this should be the record's primary key, e.g., the primary key of a Client Log.
  - **NOTE:** Certain forms, e.g., any of the log forms will simply activate an existing instance rather than open a second instance of the form if there is already an open form showing the same record.
- **page**
  - The page to go to.
  - This can use the '*' wildcard character to cater for pages like the 'Logs' page on the Clients form which has a numeric suffix that changes depending on the count of unactioned logs, e.g.:

    ```
    page=Logs*
    ```
- **forceNew**
  - A Boolean value indicating that a new form instance should be used rather than loading a record into an existing form.

In addition to the above, the following parameters are also supported:

- **closeActiveForm**
  - Indicates whether to close the active form, e.g., the form from which the Application Shortcut has been launched.
- **state**
  - Specifies the state to put the form into. One of the following:
    - Normal
    - Maximized
    - Minimized
- **left**, **top**, **width**, **height**
  - Allows the position and size of the form to be specified.

Examples of this Application Shortcut are:

```
FormShow?form=Clients

FormShow?form=Clients&id=C10000

FormShow?form=Clients&id=C10000&page=Contacts

FormShow?form=Clients&id=C10000&page=Logs*&forceNew=True

FormShow?form=TaskManager&state=Maximized
```

As of finPOWER Connect 2.02.00, the following helper method can be used to create an Application Shortcut:

```
ApplicationShortcut = finBL.CreateApplicationShortcutFormShow("Clients", "C10000", "Logs*")
```

Special Parameters

Some forms can specify special, non-standard, parameters. Some of these are listed in the table below:

Form	Parameter	Descriptiion

Create Document wizard, e.g.,
`DocumentCreateAccount`
`DocumentCreateClient`
	`id`	The Id of the record to select, e.g., the Client or Account Id.
		
		`id=L10000`

	`fileType`	The file type to filter by: WordVba ExcelVba Email Sms Script Log Html `fileType=Sms`
	`documentId`	The Id of the default Document to select. `documentId=AA`
	`limitDocumentList`	Used in conjunction with the documentId parameter, this limits the Documents grid. `limitDocumentList=True`
	`Value`	The default value, e.g., Email address or SMS phone number. `value=test@test.com`
Document Execute (when running a Document from Report Explorer), e.g.:	`id`	The Id of the Document to select by default.

DocumentsAccount DocumentsClient		id=AA
	fileType	The file type to filter by: WordVba ExcelVba Email Sms Script Log Html fileType=Sms
	primaryRange	A value to default the primary range to, e.g., the Accounts range for an Account-type Document. primaryRange=L10000,RC10001
Activate Accounting Ledgers: AccountAccountingLedgersActivate	entityId	Default value for the Entity. entityId=MAIN
Process Direct Debit Payments: AccountDirectDebitsProcess	bankAccountId	Default value for the Bank Account. bankAccountId=CHQ
Bank Account Enquiry BankAccountEnquiry	clientId	Default value for the Client. clientId=C10000
Credit Enquiry CreditEnquiry	clientId	Default value for the Client. clientId=C10000
Import Bank Transactions BankImport	bankAccountId	Default value for the Bank Account. bankAccountId=CHQ
	serviceId	Default value for the Import Service. serviceId=QIF

NOTE: If you are trying to show a form that you think accepts special parameters and it is not listed in the table above, please contact Intersoft Systems.

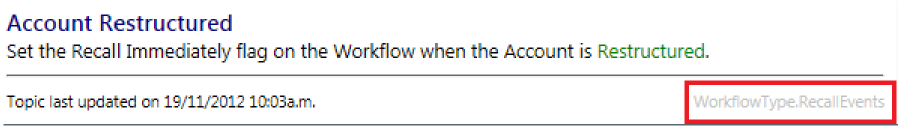
Global

Global type Application Shortcuts are currently used only for internal finPOWER Connect functionality.

HelpTopic

Show a help topic.

Each topic in the finPOWER Connect help file is identified by a Topic Id. This is visible at the bottom right of the help page, e.g.:



Account Restructured

Set the Recall Immediately flag on the Workflow when the Account is Restructured.

Topic last updated on 19/11/2012 10:03a.m.

WorkflowType.RecallEvents

This Application Shortcut has the following parameters:

- **fileName**
    - The name of the HTML Help file (a .chm file).
    - If no path is specified for the file, the finPOWER Connect program folder is assumed.
    - If omitted, the main finPOWER Connect Help file is assumed.
- **id**
    - The Id of the topic to show.

Examples of this Application Shortcut are:

```
HelpTopic?id=WorkflowType.RecallEvents
```

As of finPOWER Connect 2.02.00, the following helper method can be used to create an Application Shortcut:

```
ApplicationShortcut = finBL.CreateApplicationShortcutHelpTopic("WorkflowType.RecallEvents")
```

HtmlShow

Show HTML or URL in a separate form.

This Application Shortcut has the following parameters:

- **html**
  - The HTML to show.
  - If this does not contain an <html> tag, it is assumed to be a partial piece of HTML and, if this is a 'popup' (see below), it will be included in a full HTML document generated from the HTML Summary Page Builder.
- **url**
  - The URL to show.
  - This can be used in place of the `html` parameter if required.
- **popup**
  - A Boolean value indicating whether the HTML should be displayed in a special 'Popup' form that will be closed immediately it loses focus.
- **popupStyle**
  - A style for the 'Popup' form.
  - One of the following:
    - Note
      - Colours the HTML document background yellow.
- **title**
  - A title for the HTML viewer form.
  - If omitted, this will be either 'View HTML' or 'Note' depending on whether a popup form if being used.
- **modal**
  - Indicates whether to display the HTML viewer form modally.
  - Not applicable to popups.
- **formKey**
  - A form key to assign when showing the HTML viewer form so that it preserves form size and position between views.
  - Not applicable to popups.
- **width**
  - The width (in pixels) for the HTML viewer form.
  - Not applicable if formKey is specified.
  - **NOTE:** The HTML viewer has a built-in minimum width. Width will not be applied if it is below this.
- **height**
  - The height (in pixels) for the HTML viewer form.
  - Not applicable if formKey is specified.
  - **NOTE:** The HTML viewer has a built-in minimum height. Height will not be applied if it is below this.

Examples of this Application Shortcut are:

```
HtmlShow?html=Test HTML

HtmlShow?html=Test HTML&popup=True

HtmlShow?url=http://www.intersoft.co.nz&title=Intersoft Website
```

As of finPOWER Connect 2.02.00, the following helper methods can be used to create an Application Shortcut:

```
ApplicationShortcut = finBL.CreateApplicationShortcutHtmlShow("Test HTML")

ApplicationShortcut = finBL.CreateApplicationShortcutHtmlShowUrl("http://www.intersoft.co.nz")
```

HtmlShowFunction

Show HTML generated from a built-in function in a separate form.

The finPOWER Connect business layer contains a class named `finSummaryPageFunctions` which is accesible from Scripts via `finBL.SummaryPageFunctions`.

This class contains helper functions to execute built-in and overridden Summary Pages, e.g., a built-in version of the Account Financial Summary page is contained within finPOWER Connect but this can be overridden via Global Settings, Accounts, Scripts.

Any of these functions can be called via Script code and the majority can also be called via an HtmlShowFunction type Application Shortcut.

- **function**
  - The function name.
- **forceBuiltIn**
  - Indicates whether to use the built-in Script regardless of whether it is overridden under Global Settings or elsewhere.
- **title**
  - A title for the HTML viewer form.
- **popup**
  - A Boolean value indicating whether the HTML should be displayed in a special 'Popup' form that will be closed immediately it loses focus.
- **popupStyle**
  - A style for the 'Popup' form.
  - One of the following:
    - Note
      - Colours the HTML document background yellow.
- **modal**
  - Indicates whether to display the HTML viewer form modally.
  - Not applicable to popups.
- **formKey**
  - A form key to assign when showing the HTML viewer form so that it preserves form size and position between views.
  - Not applicable to popups.
- **width**
  - The width (in pixels) for the HTML viewer form.
  - Not applicable if formKey is specified.
  - **NOTE:** The HTML viewer has a built-in minimum width. Width will not be applied if it is below this.
- **height**
  - The height (in pixels) for the HTML viewer form.
  - Not applicable if formKey is specified.
  - **NOTE:** The HTML viewer has a built-in minimum height. Height will not be applied if it is below this.

Other parameters differ on a per-function basis and can be determined by looking at the function's signature, e.g., the `AccountFinancialSummary()` method has the following parameters:

- **accountId**

  - The Account Id.
- **index**
  - The index of the Account Calculation version to use in the summary.
  - If omitted, a value of -1 is assumed that uses the current Calculation.

Examples of this Application Shortcut are:

```
HtmlShowFunction?function=ClientWarningsSummary&clientId=C10000

HtmlShowFunction?function=AccountFinancialSummary&forceBuiltIn=True&accountId=L10035&index=-1
```

A selection of available functions is listed in the next section.

As of finPOWER Connect 2.02.00, the following helper method can be used to create an Application Shortcut:

```
Dim Parameters As ISKeyValueList

Parameters = finBL.CreateKeyValueList()
Parameters.SetString("ClientId", "C10000")

ApplicationShortcut = finBL.CreateApplicationShortcutHtmlShowFunction("ClientWarningsSummary",
False, "Warnings", Parameters)
```

Supported Functions

The following table lists some of the methods (functions) that can be called from an HtmlShowFunction Application Shortcut:

Function	Description	Parameters
`AccountFinancialSummary`	Account Financial Summary.	`accountId (String)` `index (Integer, -1)`
`AccountPaymentArrangementSummary`	Account Payment Arrangement Summary.	`accountId (String)` `index (Integer, -1)`
`AccountPaymentsSummary`	Account Payments Summary.	`accountId (String)`
`AccountStandardTransactionSummary`	Account Standard Transaction Summary.	`accountId (String)` `index (Integer)`
`AccountTransactionSummary`	Account Transaction Summary.	`accountId (String)` `index (Integer)`
`AccountWarningsSummary`	Account Warnings Summary.	`accountId (String)`
`ClientWarningsSummary`	Client Warnings Summary.	`clientId (String)`
`SecurityStmtWarningsSummary`	Security Statement Warnings Summary.	`securityStmtId (String)`

NOTE: Only methods in `finBL.SummaryPageFunctions` that do not contain parameters of Object types (e.g., `finAccount`) can be run from an Application Shortcut and the above table shows only a subset of the available functions.

HtmlShowScriptEvent

Shows HTML generated from a Script event.

See the Script Events section for more information.

This Application Shortcut has the following parameters:

- **scriptId**
  - The Id of the Summary Page (version 2) type Script.
- **eventId**
  - The Event Id to pass to the Script.
- **title**
  - A title for the HTML viewer form.
  - If omitted, this will be 'View HTML'.
- **modal**
  - Indicates whether to display the HTML viewer form modally.
  - Not applicable to popups.
- **formKey**
  - A form key to assign when showing the HTML viewer form so that it preserves form size and position between views.
  - Not applicable to popups.
- **width**
  - The width (in pixels) for the HTML viewer form.
  - Not applicable if formKey is specified.
  - **NOTE:** The HTML viewer has a built-in minimum width. Width will not be applied if it is below this.
- **height**
  - The height (in pixels) for the HTML viewer form.
  - Not applicable if formKey is specified.
  - **NOTE:** The HTML viewer has a built-in minimum height. Height will not be applied if it is below this.

All other parameters are passed to the Script via the `eventArgs` Key/ Value List parameter.

Examples of this Application Shortcut are:

```
HtmlShowScriptEvent?scriptId=TEST&eventId=ShowSummary

HtmlShowScriptEvent?scriptId=TEST&eventId=ShowSummary&accountId=L10000
```

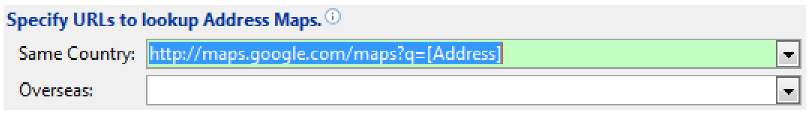
As of finPOWER Connect 2.02.00, the following helper method can be used to create an Application Shortcut:

```
Dim Parameters As ISKeyValueList

Parameters = finBL.CreateKeyValueList()
Parameters.SetString("AccountId", "L10000")

ApplicationShortcut = finBL.CreateApplicationShortcutScriptEvent("TEST", "ShowSummary",
Parameters)
```

MapShow

Show a map for an address.

This simply generates a URL for an external service such as Google Maps to use. The URL (and therefore the service used) can be defined under Global Settings, Clients, Addressing, e.g.:


Specify URLs to lookup Address Maps.

Same Country: http://maps.google.com/maps?q=[Address]

Overseas:

This Application Shortcut has the following parameters:

- **address**
  - The address to show formatted as a single line.
    - Most services require that the address contains the country. Also, for New Zealand addresses, it seems to be safer to omit the suburb and only include the city.
  - If omitted, the current address from the `source` parameter (below) will be used.
  - **NOTE:** Using one of the `finBL.CreateApplicationShortcutMapShow()` helper methods removes the complexity of formatting the address correctly.
- **source**
  - If an address is not specified, then `source` can be set to one of the following:
    - Account
    - Client
- **sourceId**
  - The Id of the source record, e.g., a Client Id or AccountId.
- **sourceType**
  - The type of the source address. One of the following:
    - Physical
    - Postal
  - If omitted, 'Physical' will be used.
- **sourceItemPk**
  - The primary key of the source item, e.g., a Client Contact Address record.

Examples of this Application Shortcut are:

```
MapShow?source=Client&sourceId=C10000

MapShow?address=19 Marine Parade, Napier, New Zealand
```

As of finPOWER Connect 2.02.00, the following helper methods can be used to create an Application Shortcut:

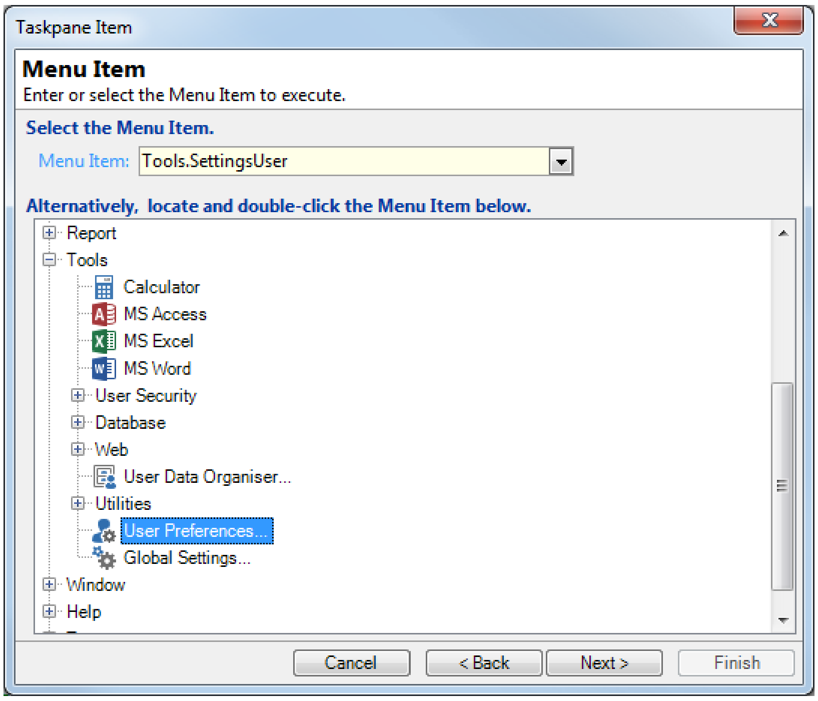
```
ApplicationShortcut = finBL.CreateApplicationShortcutMapShowClient("C10000", True)

ApplicationShortcut = finBL.CreateApplicationShortcutMapShowAccount("L10000")

ApplicationShortcut = finBL.CreateApplicationShortcutMapShowClientContactAddress("C10000", 79)

ApplicationShortcut = finBL.CreateApplicationShortcutMapShow("19 Marine Parade", "", "Napier")
```

MenuCommand

Call a menu command.

Each item on the finPOWER Connect menu is assigned a unique Id. To determine the Id of a particular menu item, do the following:

- Right-click in the Taskpane and select **Add...**
- In the Taskpane Item wizard, select an Application Shortcut type of **Menu Item**.
- Click **Next >>**.
- The **Menu Item** page shows a treeview representing the finPOWER Connect menu. Double-clicking an item will copy the Id of the menu item to the 'Menu Item' dropdown, e.g.:


Taskpane Item

Menu Item

Enter or select the Menu Item to execute.

Select the Menu Item.

Menu Item: Tools.SettingsUser

Alternatively, locate and double-click the Menu Item below.

This Application Shortcut has the following parameters:

- **id**
  - The Id of the menu command to show.

Examples of this Application Shortcut are:

```
MenuCommand?id=Tools.SettingsUser
```

As of finPOWER Connect 2.02.00, the following helper method can be used to create an Application Shortcut:

```
ApplicationShortcut = finBL.CreateApplicationShortcutMenuCommand("Tools.SettingsUser")
```

PageSet

Run a Page Set.

Page Sets are fully documented in the **finPOWER Connect 2 Page Sets** document which can be downloaded from the Intersoft Website.

This Application Shortcut has the following parameters:

- **id**
  - The Id of the Page Set to run.
- **visiblePageCodes**
  - An optional, comma-separated list of Page Codes to display.
  - This is not applicable to 'Wizard' type Page Sets.
- **navigationMethod**
  - Each Page Set defines a Navigation Method, e.g., 'Wizard' or 'Tabbed Pages'.
  - Specifying this parameter allows the Navigation Method to be overridden, e.g., to display a 'Wizard' type Page Set as 'Tabbed Pages'. This should be one of the following:
    - Wizard
    - TabbedPages
    - SinglePage
- **modal**
  - A Boolean value indicating whether to display this Page Set as a Modal form, i.e., the User cannot interact with an other finPOWER Connect forms or menus when the Page Set form is displayed.
- **pseudoModal**
  - A Boolean value indicating whether to display this Page Set as a 'Pseudo-Modal' form, i.e., the User cannot interact with the parent form (e.g., the form containing the Summary Page from which this Page Set was run) when the Page Set form is displayed.
- **allowInactive**
  - A Boolean value indicating whether to show a Page Set if the record is flagged as 'Inactive'.

NOTE: Any other parameters specified as part of the Application Shortcut will be passed to the Page Set and will be available from the Page Set Script via the `psh.Parameters` Key/Value List.

Examples of this Application Shortcut are:

```
PageSet?id=LOANAPP

PageSet?id=LOANAPP&navigationMethod=TabbedPages

PageSet?id=LOANAPP&navigationMethod=TabbedPages&visiblePageCodes=START,LOAN,CLIENT1
```

As of finPOWER Connect 2.02.00, the following helper method can be used to create an Application Shortcut:

```
ApplicationShortcut = finBL.CreateApplicationShortcutPageSet("LOANAPP",
isefinPageSetNavigationMethod.TabbedPages)
```

PhoneDial

Dial a phone number.

This uses the built-in Windows functionality to attempt to dial a phone number.

This Application Shortcut has the following parameters:

- **phoneNumber**
  - The phone number to call.
- **name**
  - The name of the person or company being called.
  - Passed to the Windows Phone Dialer API.
- **notes**
  - Notes regarding the call.
  - Passed to the Windows Phone Dialer API.

Examples of this Application Shortcut are:

```
PhoneDial?phoneNumber=068355237

PhoneDial?phoneNumber=068355237&name=Intersoft&name=Calling Napier office
```

As of finPOWER Connect 2.02.00, the following helper method can be used to create an Application Shortcut:

```
ApplicationShortcut = finBL.CreateApplicationShortcutPhoneDial("068355237")
```

Queue

Run a Queue.

This Application Shortcut has the following parameters:

- **id**
  - The Id of the Queue.
- **autoRun**
  - A Boolean value indicating whether to automatically run the Queue.
  - If this is not `True`, the Run Queue wizard will be displayed but the Queue will not be started.

NOTE: Any other parameters specified as part of the Application Shortcut will be used to update the Queue's parameters.

Examples of this Application Shortcut are:

```
Queue?id=Reports1

Queue?id=Reports1&DateAsAt=8/21/2014&autoRun=True
```

As of finPOWER Connect 2.02.00, the following helper method can be used to create an Application Shortcut:

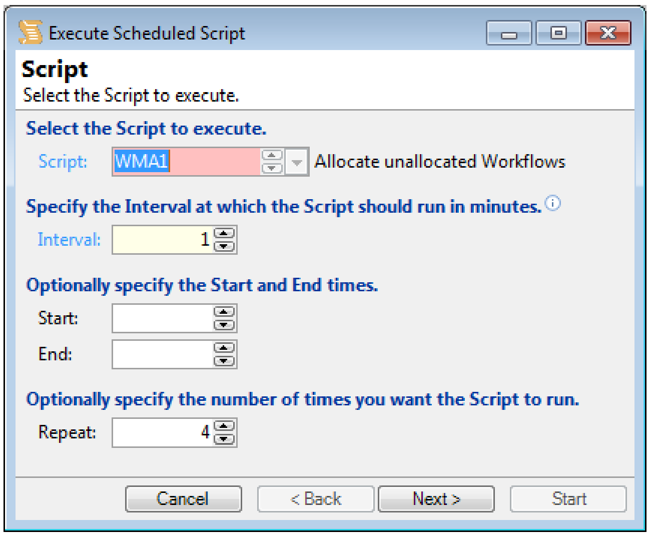
```
Dim Parameters As ISKeyValueList

Parameters = finBL.CreateKeyValueList()
Parameters.SetDate("DateAsAt", Now.Date)

ApplicationShortcut = finBL.CreateApplicationShortcutQueue("Test", True, Parameters)
```

ScheduledScript

Run a scheduled Script.

A Scheduled Script can only be run when no other forms are open in finPOWER Connect. Therefore, this type of Application Shortcut cannot be run from a Summary Page and is more suited to a shortcut in the Taskpane or as a startup Application Shortcut for a User (defined user User Preferences, Tasks & Workflows, Start Up/ Shut Down).

The Script is run via the special Execute Schedule Script wizard, e.g.:


Execute Scheduled Script
Script
Select the Script to execute.
Select the Script to execute.
Script: WMA1 Allocate unallocated Workflows
Specify the Interval at which the Script should run in minutes.
Interval: 1
Optionally specify the Start and End times.
Start:
End:
Optionally specify the number of times you want the Script to run.
Repeat: 4
Cancel
< Back
Next >
Start

This Application Shortcut has the following parameters:

- **id**
  - The Id of the Script. The Script must be one of the following Script types:
    - General
    - General (User Interface)
    - HTML Report
    - Summary Page or Summary Page (version 2)
- **autoRun**
  - A Boolean value indicating whether to automatically start the Execute Scheduled Script wizard executing.
  - If this is not `True`, the wizard will be displayed but the not started.
- **intervalMinutes**
  - The interval at which the Script should run in minutes.
  - It not specified, this will default to 10.
- **startTime**, **endTime**
  - The optional Start and End times.
  - If specified, these should be dates containing a time portion (the actual date will be ignored) and should be formatted as **M/d/yyyy HH:mm:ss**, e.g., **1/1/2000 14:30**.
- **repeatCount**
  - Optionally, the number of times the Script should be run.

Examples of this Application Shortcut are:

```
ScheduledScript?id=TEST&startTime=1/1/2000 14:30&endTime=1/1/2000 17:30&autoRun=True
```

As of finPOWER Connect 2.02.00, the following helper method can be used to create an Application Shortcut:

```
ApplicationShortcut = finBL.CreateApplicationShortcutScheduledScript("TEST", True, 10, New
DateTime(2000, 1, 1, 14, 30, 0),  New DateTime(2000, 1, 1, 17, 30, 0))
```

Script

Run a Script.

This Application Shortcut has the following parameters:

- **id**
  - The Id of the Script. The Script must be one of the following Script types:
    - General
    - General (User Interface)
    - HTML Report
    - Summary Page or Summary Page (version 2)
  - **NOTE:** Action type Scripts are not run from this type of Application Shortcut and are detailed in the Action Scripts section below.
- **targetObjectId**
  - This is only applicable to Summary Page and Summary Page (version 2) type Scripts and will be passed across as the `source` parameter to the Script.
  - **NOTE:** The Summary Page Script must handle this correctly for this to be of any use, e.g., the built-in Account Key Details Script can receive a `source` which is a `finAccount` object or a String representing an Account Id.
- **showUi**
  - A Boolean value indicating whether to show the User Inteface (where applicable), e.g., for General type Scripts, if this is set to `True` (the default), the Execute Script wizard will be opened.
  - **NOTE:** This only applies to General and General (User Interface) type Scripts.
- **autoRun**
  - A Boolean value indicating whether to automatically start the Execute Script wizard executing.
  - If this is not `True`, the wizard will be displayed but the not started.
  - **NOTE:** This only applies to General and General (User Interface) type Scripts.

NOTE: Any other parameters specified as part of the Application Shortcut will be used to update the Script's parameters where applicable.

If the Execute Script wizard is being displayed then the parameters defined on the Script will be updated from the Application Shortcut's parameters; any parameters not defined on the Script will be ignored.

If no User Interface is being displayed then the Script's Parameters will be updated from the Application Shortcut's parameters AND, any parameters defined in the Application Shortcut that are not defined as parameters on the Script will be added and will be available to the Script.

Examples of this Application Shortcut are:

```
Script?id=TEST&autoRun=True

Script?id=SP.ACCOUNT&targetObjectId=L1000
```

As of finPOWER Connect 2.02.00, the following helper methods can be used to create an Application Shortcut:

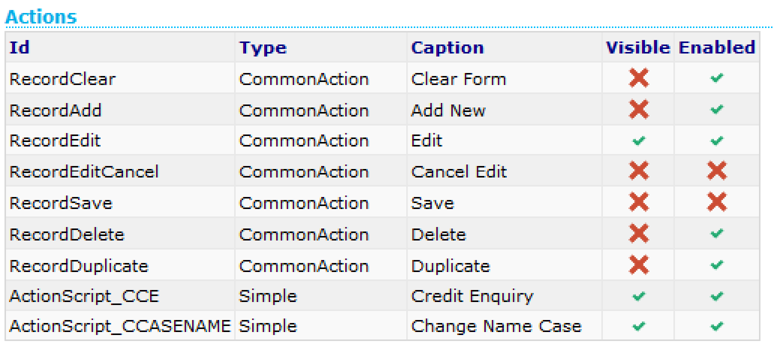
```
ApplicationShortcut = finBL.CreateApplicationShortcutScript("SP.ACCOUNT", "L10035")

Dim Parameters As ISKeyValueList

Parameters = finBL.CreateKeyValueList()
Parameters.SetString("Name", "Intersoft")

ApplicationShortcut = finBL.CreateApplicationShortcutScript("TEST", True, True, Parameters)
```

Action Scripts

Action Scripts cannot be run from 'Script' type Application Shortcuts.

Action Scripts exist as Form Actions on the relevant form, e.g., if you have a Client Action Script, this will show as a special Action on the Clients form.

Each Action Script has a unique Id, e.g., viewing the Form Details report for the Clients form:



Actions

Id	Type	Caption	Visible	Enabled
RecordClear	CommonAction	Clear Form	✗	✓
RecordAdd	CommonAction	Add New	✗	✓
RecordEdit	CommonAction	Edit	✓	✓
RecordEditCancel	CommonAction	Cancel Edit	✗	✗
RecordSave	CommonAction	Save	✗	✗
RecordDelete	CommonAction	Delete	✗	✓
RecordDuplicate	CommonAction	Duplicate	✗	✓
ActionScript_CCE	Simple	Credit Enquiry	✓	✓
ActionScript_CCASENAME	Simple	Change Name Case	✓	✓

Shows two actions represented by Action type Scripts:

- ActionScript_CCE
- ActionScript_CCASENAME

NOTE: The part after the 'ActionScript_' prefix is the Script's Id, therefore the Action Id of an Action Script is always 'ActionScript_*ScriptId*'.

Therefore, to execute an Action Script, you would use a 'FormAction' type Application Shortcut, e.g.:

```
FormAction?action=ActionScript_CCE
```

Any parameters specified as part of the Application Shorcut are available to the Action Script via the `ScriptInfo.Properties` Key/ Value List, e.g., the following Application Shortcut:

```
FormAction?action=ActionScript_CCE&Value=Proper
```

Calls Script 'CCE' which can then access the custom 'Value' parameter as follows:

```
Value = ScriptInfo.Properties.GetString("Value")
```

ScriptEvent

Run a Script event.

See the Script Events section for more information.

This Application Shortcut has the following parameters:

- **scriptId**
  - The Id of the Summary Page (version 2) type Script.
- **eventId**
  - The Event Id to pass to the Script.
- **title**
  - A title for the HTML viewer form (if applicable).
  - If omitted, this will be 'View HTML'.

All other parameters are passed to the Script via the `eventArgs` Key/ Value List parameter.

Examples of this Application Shortcut are:

```
ScriptEvent?scriptId=TEST&eventId=AddWorkflow&accountId=L10000
```

As of finPOWER Connect 2.02.00, the following helper method can be used to create an Application Shortcut:

```
Dim Parameters As ISKeyValueList

Parameters = finBL.CreateKeyValueList()
Parameters.SetString("AccountId", "L10000")

ApplicationShortcut = finBL.CreateApplicationShortcutScriptEvent("TEST", "AddWorkflow",
Parameters)
```

WebServicesToken

Launch an external Web Application which uses the finPOWER Connect Web Services and pass it a special token that can be used to connect to the Web Services without the current User needing to re-sign-in.

This Application Shortcut has the following parameters:

- **url**
  - The URL of a page in the external Web Application that can use a token to connect to the Web Services.
  - **NOTE:** This must contain a '[token]' smart tag that will be replaced with the token generated by finPOWER Connect when the Application Shortcut is executed.
- **openInternal**
  - Set to `True` to open the Web Application in an HTML Viewer from rather than an external Web browser.
- **title**
  - A title for the HTML Viewer form (if applicable, i.e., openInternal is `True`).
  - If omitted, this will be 'Web Services Application'.
- **openInSource**
  - Set to `True` to open the Web Application in the same Web Browser controller from which the Application Shortcut has been clicked.

An example of this Application Shortcut is:

```
WebServicesToken?url=https%3A%2F%2Fwww.yourwebapplication.com%2Ftest.aspx%3Fwst%3D%5Btoken%5D&
openInternal=true&title=Your Web Application
```

The following helper method can be used to create an Application Shortcut:

```
Url = "https:www.yourwebapplication.com/test.aspx?wst=[token]"
ApplicationShortcut = finBL.CreateApplicationShortcutWebServicesToken(Url, False, False "My App")
```

Overriding Built-In Blocks

Most built-in Summary Pages contain blocks that list one or more Summary Tables.

These Summary Tables are generated internally via the `finBL.SummaryPageStandardBlocks` property. Each of the methods of this class is used to retrieve one or more Summary Tables for a built-in block, e.g.:

```
' Workflows
sb.BlockBegin("Workflows")
sb.SummaryTablesAppend(finBL.SummaryPageStandardBlocks.Client_IncompleteWorkflows(ScriptInfo,
mClient, , ScriptInfo.FormRecordMode = iseFormRecordMode.Loaded))
sb.BlockEnd()
```

Each method in the SummaryPageStandardBlocks class can be overridden (described later in this section) but a Script can always force the built-in Summary Tables to be returned by specifying a parameter of `True` when accessing the `SummaryPageStandardBlocks` property, e.g.:

```
' Workflows
sb.BlockBegin("Workflows")
sb.SummaryTablesAppend(finBL.SummaryPageStandardBlocks(True).Client_IncompleteWorkflows(ScriptInfo
, mClient, , ScriptInfo.FormRecordMode = iseFormRecordMode.Loaded))
sb.BlockEnd()
```

Overriding a built-in method allows the following to be achieved, centrally, without having to modify the built-in finPOWER Connect Summary Pages:

- Replace the contents of a standard block.
  - E.g., you may wish to completely replace the content of the 'Summary' block displayed on the Client Key Details page.
- Add or remove rows from a Summary Table.
  - E.g., Add a row detailing a Client's Branch to the 'Summary' block displayed on the Client Key Detail page.
- Add a whole new Summary Table to a standard block.
  - Most methods in `finBL.SummaryPageStandardBlocks` return a collection of Summary Tables, therefore it is easy to add a new Summary Table which will then be rendered in the block.
  - E.g., Add an 'Employer Details' Summary Table to the bottom of the 'Summary' block displayed on the Client Key Details page.

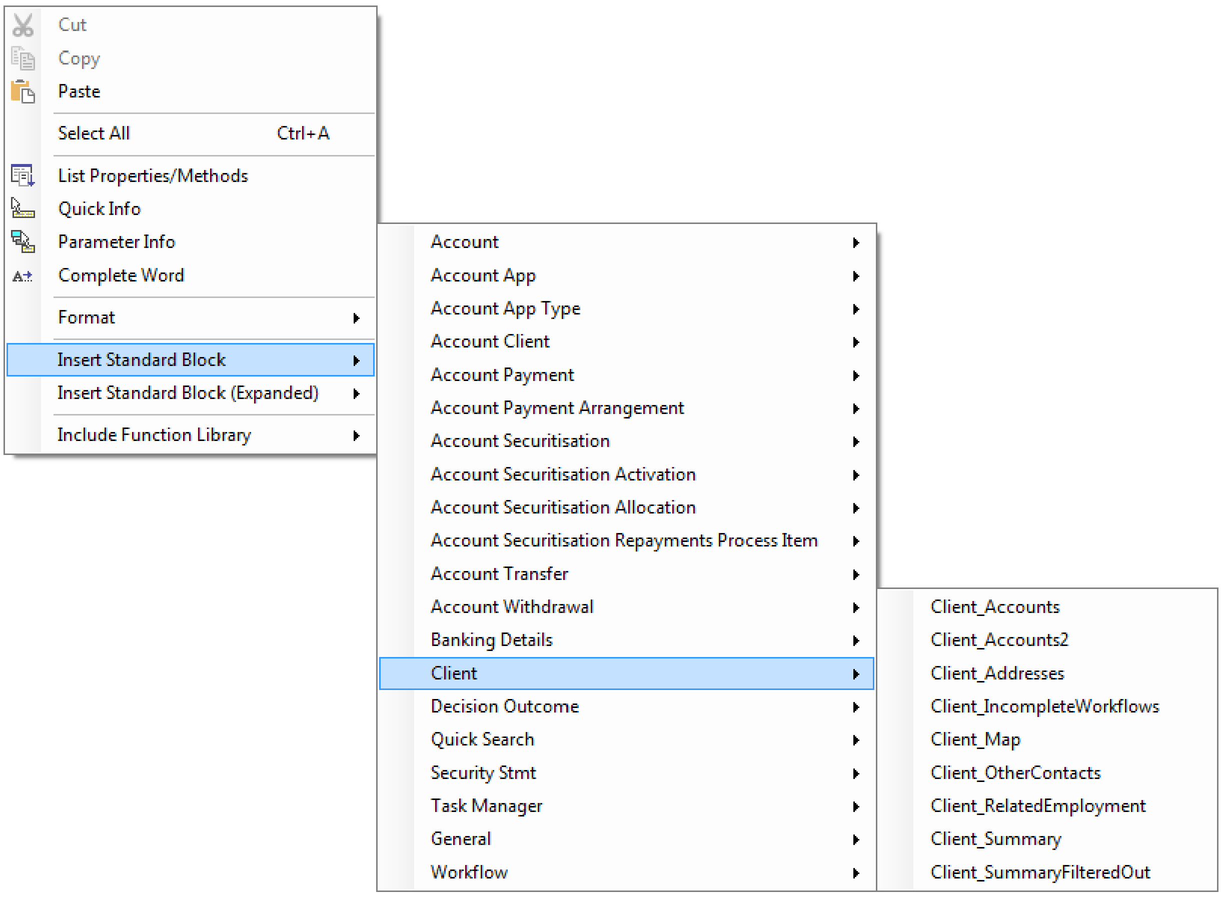
WARING: If the Script attempting to override a built-in block errors for any reason, the built-in code will be used to generate the block and a Trace warning will occur. This can be viewed from the **Tools**, **Utilities**, **Debug Window** form.

Summary Page Standard Block Override Type Script

Every finPOWER Connect database can define a single 'Summary Page Standard Block Override' which can override any built-in Summary Tables.

This type of Script is different from most other Script types in that it does not define a 'Main' method. Instead, this Script inherits from the built-in `finSummaryPageStandardBlocksBase` class.

Right-clicking in the Script editor for this type of Script displays a menu listing all methods that can be overridden, e.g.:



The two options available when overriding a standard block are 'Insert Standard Block' and 'Insert Standard Block (Expanded)'.

Insert Standard Block

This option inserts a function that simply calls the Base (i.e., built-in) functionality, e.g.:

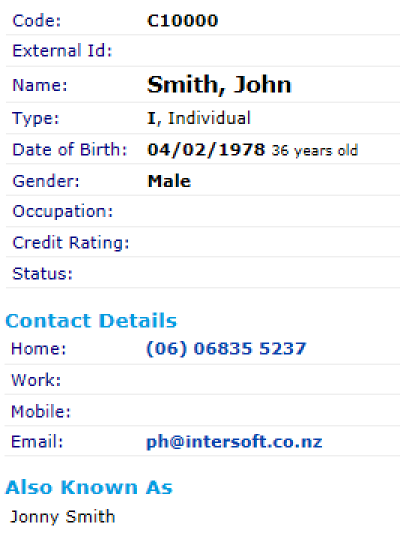
```
Public Overrides Function Client_Summary(callerScriptInfo As ISScriptExecutionInfo,
                                         client As finClient,
                                Optional includeClientIdHyperlink As Boolean = True,
                                Optional includeAkas As Boolean = True,
                                Optional ByRef includedContactMethodPks As ISList = Nothing,
                                Optional otherParameters As ISKeyValueList = Nothing) As
ISSummaryTables

  Client_Summary = MyBase.Client_Summary(callerScriptInfo, client, includeClientIdHyperlink,
includeAkas, includedContactMethodPks, otherParameters)

End Function
```

This allows you to modify the internally built Summary Tables collection, e.g., to add or delete rows or to add an entirely new Summary Table.

The call to the `MyBase.Client_Summary()` method will always return the Summary Tables generated from the most recent built-in code. The remarks detail the names of the Summary Tables returned from this method, e.g.:

```
'''  <para>
'''   Returns the following Summary Tables:
'''   <list>
'''    <item><code>Main</code><description>Main details.</description></item>
'''    <item><code>Contacts</code><description>Contact details.</description></item>
'''    <item><code>Akas</code><description>AKA details (optional).</description></item>
'''   </list>
'''  </para>
```

These correspond the logical blocks within the Client Key Details Summary, e.g.:



Code:	**C10000**
External Id:	
Name:	**Smith, John**
Type:	**I**, Individual
Date of Birth:	**04/02/1978** 36 years old
Gender:	**Male**
Occupation:	
Credit Rating:	
Status:	

Contact Details

Home:	**(06) 06835 5237**
Work:	
Mobile:	
Email:	**ph@intersoft.co.nz**

Also Known As

Jonny Smith

Example code to modify these built-in Summary Tables is given later in this section.

NOTE: This is a good starting point for completely replacing a built-in block. Simply insert the standard block and then remove the call to the base-class method (E.g., `Client_Summary = MyBase.XXX`). Create and return your own `ISSummaryTables` collection.

Insert Standard Block (Expanded)

This option inserts a function that contains the actual built-in code, e.g.:

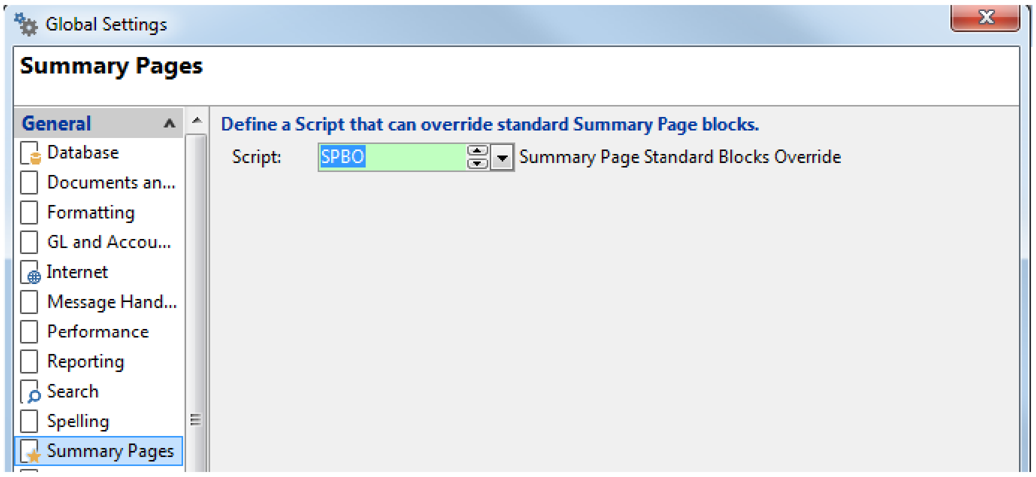
```
Public Overrides Function Client_Summary(callerScriptInfo As ISScriptExecutionInfo,
                                          client As finClient,
                                Optional includeClientIdHyperlink As Boolean = True,
                                Optional includeAkas As Boolean = True,
                                Optional ByRef includedContactMethodPks As ISList = Nothing,
                                Optional otherParameters As ISKeyValueList = Nothing) As
ISSummaryTables

  Dim ClientAka As finClientAka
  Dim ClientContactMethod As finClientContactMethod
  Dim ClientContactMethodHome As finClientContactMethod
  Dim ClientContactMethodMobile As finClientContactMethod
  Dim ClientContactMethodWork As finClientContactMethod
  Dim LinkedRecordDescription As String
  Dim LinkedRecordId As String
  Dim strTemp As String
  Dim SummaryTableAkas As ISSummaryTable
  Dim SummaryTableContacts As ISSummaryTable
  Dim SummaryTableMain As ISSummaryTable
  Dim SummaryTables As ISSummaryTables

  ' Initialise ByRef Parameters
  includedContactMethodPks = New ISList()

  ' Get Constants

  ' Initialise
  If callerScriptInfo Is Nothing Then callerScriptInfo = finBL.CreateScriptExecutionInfo()
  If otherParameters Is Nothing Then otherParameters = finBL.CreateKeyValueList()
  SummaryTableAkas = finBL.CreateSummaryTable()
  SummaryTableContacts = finBL.CreateSummaryTable()
  SummaryTableMain = finBL.CreateSummaryTable()
  SummaryTables = finBL.CreateSummaryTables()
```

This allows you view the internal code which is useful for seeing examples of where certain information comes from or how a Summary Table is formatted.

Configuration

Every finPOWER Connect database can define a single 'Summary Page Standard Block Override' which can override any built-in Summary Tables.

The database can then be configured to use this Script. This is done via the Global Settings, General, Summary Pages page:


Global Settings

Summary Pages

Define a Script that can override standard Summary Page blocks.

Script: SPBO Summary Page Standard Blocks Override

Adding or Deleting Rows to Existing Summary Tables

The following example updates the Main table in the Client Summary block as follows:

- Remove the 'Type' row detailing the Client Type.
- Add a new 'Branch' row beneath the 'Name' row.

```
Public Overrides Function Client_Summary(callerScriptInfo As ISScriptExecutionInfo,
                                         client As finClient,
                                Optional includeClientIdHyperlink As Boolean = True,
                                Optional includeAkas As Boolean = True,
                                Optional ByRef includedContactMethodPks As ISList = Nothing,
                                Optional otherParameters As ISKeyValueList = Nothing) As
ISSummaryTables

  Dim SummaryTables As ISSummaryTables
  Dim SummaryTableMain As ISSummaryTable

  ' Get Built-In Summary Tables
  SummaryTables = MyBase.Client_Summary(callerScriptInfo, client, includeClientIdHyperlink,
includeAkas, includedContactMethodPks, otherParameters)

  ' Update 'Main' table
  If SummaryTables.Exists("Main") Then
    SummaryTableMain = SummaryTables("Main")

    ' Remove 'Type'
    SummaryTableMain.Rows.RemoveByCaption("Type:")

    ' Add 'Branch' after 'Name'
    With SummaryTableMain.Rows.InsertAfter("Name:").Cells
      .AddCaption("Branch:")
      .AddCodeDescription(client.BranchId, finBL.Branches)
    End With
  End If

  Return SummaryTables

End Function
```

Adding or Deleting an Existing Summary Table

'TODO:

Script Events

At a basic level, Summary Page (version 2) Scripts receive one or more parameters and return an HTML document.

However, looking at the function signature, you will see that it accepts parameters for `eventId` and `eventArgs`, e.g.:

```
Public Function Main(source As Object,
                     eventId As String,
                     eventArgs As ISKeyValueList,
                     ByRef handled As Boolean,
                     ByRef returnValues As ISKeyValueList,
                     ByRef text As String) As Boolean

  ' Assume Success
  Main = True

  ' Handle Events
  Select Case eventId
    Case ""
      ' Main event, i.e., return main page content
      text = "My HTML"
  End Select

End Function
```

As you can see from the above example, the Script has a `Select Case eventId` and, only when `eventId` is blank, does the Script return any HTML.

Summary Page Scripts can be run with no `eventId` (the default) but they can also be run with the specified `eventId` and any additional parameters that the Script might require included in the `eventArgs` Key/ Value List.

The following are possible uses for Script events:

- Provide a hyperlink in the main Summary Page which, when clicked, displays a popup HTML form displaying additional details not shown on the main page.
  - The PopupNote HTML Template can achieve this to a degree but has the following disadvantages:
    - It is only designed to show small amounts of information.
    - The HTML defined for the popup note must be generated with the main Summary Page rather than on-demand, i.e., when the link is clicked.
  - This can be achieved efficiently using Script Events in conjunction with an HTMLShowScriptEvent Application Shortcut.
- Provide a hyperlink in the main Summary Page which, when clicked, executes Script code to start a Workflow for an Account.
  - This can be achieved using Script Events in conjunction with a ScriptEvent Application Shortcut.

The above uses are discussed in the next sections.

When using Script Events, it is important to understand that the Script is being re-run each time an event is called, therefore, any parameters passed into the Script originally such as `source` will not be available when the Script Event is called.

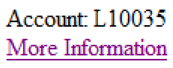
Therefore, when calling the Script Event, generally via an Application Shortcut, you must supply enough information to the Script via the `eventArgs` parameter. E.g., if the main Summary Page displayed an Account Summary, you will probably want to pass the Account's Id to the Script Event if the event requires access to the same Account record.

Showing a Popup HTML Form

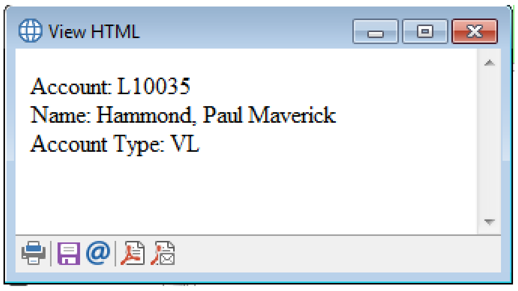
The example in this section shows a very simple, unformatted HTML page (created manually rather than using the HTML Summary Page Builder for conciseness).

The HTML displays information about an Account and provides a hyperlink to display more details. It assumes that this is an Account Summary Page that appears on the Accounts form and is therefore passed a `finAccount` object in the `source` parameter.

When run, the sample produces a main Summary Page, e.g.:


Account: L10035
More Information

When the hyperlink is clicked the SAME Script is re-called (via an HtmlShowScriptEvent Application Shortcut) passing in an `eventId` of 'MoreInfo' and produces the following:



Account: L10035
Name: Hammond, Paul Maverick
Account Type: VL

Note the highlighted parts in the sample Script below:

```
' Objects
Private mAccount As finAccount

Public Function Main(source As Object,
                     eventId As String,
                     eventArgs As ISKeyValueList,
                     ByRef handled As Boolean,
                     ByRef returnValues As ISKeyValueList,
                     ByRef text As String) As Boolean

  ' Assume Success
  Main = True

  ' Initialise
  If source IsNot Nothing Then
    mAccount = DirectCast(source, finAccount)
  End If

  ' Handle Events
  Select Case eventId
    Case ""
      ' Main event, i.e., return main page content
      Main = GetPage(text)

    Case "MoreInfo"
      ' More information event
      Main = MoreInfo(eventArgs.GetString("AccountId"), text)
  End Select

End Function

Private Function GetPage(ByRef html As String) As Boolean

  Dim ApplicationShortcut As ISApplicationShortcut

  ' Assume Success
  GetPage = True

  ' Create Application Shortcut to call Script Event
  ApplicationShortcut = finBL.CreateApplicationShortcut()
  With ApplicationShortcut
    .Action = "HtmlShowScriptEvent"

    With .Parameters
```

```
      ' Mandatory Parameters
      .SetString("scriptId", ScriptInfo.ScriptId)
      .SetString("eventId", "MoreInfo")

      ' Event-specific Parameters
      .SetString("accountId", mAccount.AccountId)
    End With
  End With

  ' Create HTML
  html = "Account: " & finBL.HtmlEncode(mAccount.AccountId) & "<br/>"
  html &= String.Format("<a href='{0}'>More Info</a>", ApplicationShortcut.ToUrlString(True))

End Function

Private Function MoreInfo(accountId As String,
                          ByRef html As String) As Boolean

  Dim Account As finAccount

  ' Assume Success
  MoreInfo = True

  ' Load Account
  Account = finBL.CreateAccount()
  MoreInfo = Account.Load(accountId)

  ' Create HTML
  html = "Account: " & finBL.HtmlEncode(Account.AccountId) & "<br/>"
  html &= "Name: " & finBL.HtmlEncode(Account.Name) & "<br/>"
  html &= "Account Type: " & finBL.HtmlEncode(Account.AccountTypeId) & "<br/>"

End Function
```

NOTE: Although this example calls an event in the same Script, it is possible to call an event in another Script by changing the `scriptId` parameter in the Application Shortcut.

If you do reference a different Script, it is recommended that you define the Script's Id as a constant rather than hardcoding it into the Application Shortcut.

Important points to take away from this example are:

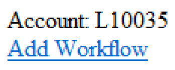
- The `Select Case` in the `Main()` function handles a new 'MoreInfo' event.
  - Each event is handled in a separate function for readability.
- The Application Shortcut created in the `GetPage()` function:
  - Uses `ScriptInfo.ScriptId` instead of hard-coding the Script's Id.
  - Adds a parameter of `AccountId` so that the MoreInfo event knows which Account to summarise.
    - All Application Shortcut parameters apart from ScriptId, EventId and a few others mentioned in the HtmlShowScriptEvent section will be passed to the Script in the `eventArgs` parameter.

Executing Script Code

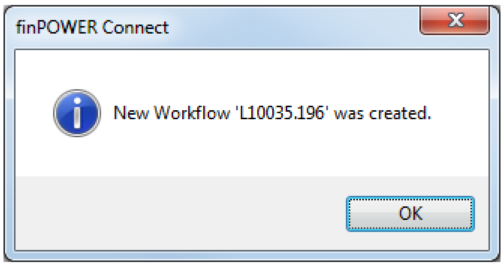
Executing Script code follows the same pattern as when showing a popup HTML form. Therefore, this section does not cover the mechanism in as much details. The only real difference is that an HTML viewer is not automatically opened upon completion of the event.

The example in this section shows a very simple, unformatted HTML page which displays information about an Account and provides a link to start a Workflow for the Account. It assumes that this is an Account Summary Page that appears on the Accounts form and is therefore passed a `finAccount` object in the `source` parameter.

When run, the Script produces a main Summary Page, e.g.:

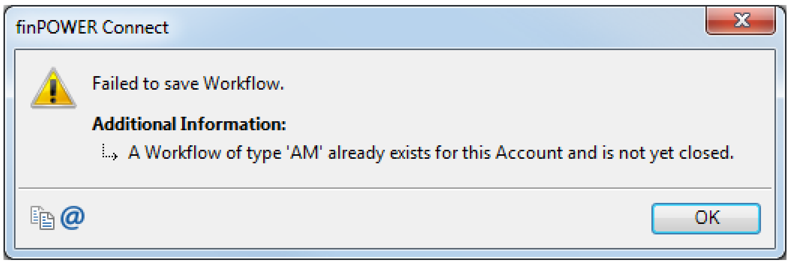

Account: L10035
Add Workflow

When the hyperlink is clicked the SAME Script is re-called (via a ScriptEvent Application Shortcut) passing in an `eventId` of 'AddWorkflow' and produces the following message:


finPOWER Connect

New Workflow 'L10035.196' was created.

Or an error, e.g.:


finPOWER Connect

Failed to save Workflow.

Additional Information:

A Workflow of type 'AM' already exists for this Account and is not yet closed.

Note the highlighted parts in the sample Script below:

```
' Objects
Private mAccount As finAccount

Public Function Main(source As Object,
                     eventId As String,
                     eventArgs As ISKeyValueList,
                     ByRef handled As Boolean,
                     ByRef returnValues As ISKeyValueList,
                     ByRef text As String) As Boolean

  ' Assume Success
  Main = True

  ' Initialise
  If source IsNot Nothing Then
    mAccount = DirectCast(source, finAccount)
  End If

  ' Handle Events
  Select Case eventId
    Case ""
      ' Main event, i.e., return main page content
      Main = GetPage(text)

    Case "AddWorkflow"
      ' Add Workflow event
      Main = AddWorkflow(eventArgs.GetString("AccountId"), text)
  End Select

End Function

Private Function GetPage(ByRef html As String) As Boolean

  Dim ApplicationShortcut As ISApplicationShortcut

  ' Assume Success
  GetPage = True

  ' Create Application Shortcut to call Script Event
  ApplicationShortcut = finBL.CreateApplicationShortcut()
  With ApplicationShortcut
    .Action = "ScriptEvent"

    With .Parameters
      ' Mandatory Parameters
      .SetString("scriptId", ScriptInfo.ScriptId)
      .SetString("eventId", "AddWorkflow")

      ' Event-specific Parameters
      .SetString("accountId", mAccount.AccountId)
    End With
  End With

  ' Create HTML
  html = ""
  html &= "Account: " & finBL.HtmlEncode(mAccount.AccountId) & "<br/>"
  html &= String.Format("<a href='{0}'>Add Workflow</a>", ApplicationShortcut.ToUrlString(True))

End Function

Private Function AddWorkflow(accountId As String,
                             ByRef message As String) As Boolean

  Dim Account As finAccount
  Dim Workflow As finWorkflow

  ' Assume Success
  AddWorkflow = True

  ' Load Account
  Account = finBL.CreateAccount()
  AddWorkflow = Account.Load(accountId)

  ' Create Workflow
  If Account.DateClosed = Nothing Then
    ' Create Workflow
    Workflow = finBL.CreateWorkflow()

    ' Initialise
    Workflow.AccountId = accountId
    AddWorkflow = Workflow.Initialise("AM")

    ' Save
    If AddWorkflow Then
      AddWorkflow = Workflow.Save()
    End If

    ' Success Message
    If AddWorkflow Then
      message = String.Format("INFO|New Workflow '{0}' was created.", Workflow.WorkflowId)
    Else
      ' Returning False will display the error message when event complete
    End If
  Else
    ' Closed (could also set an Error and return False)
    message = "WARNING|Could not add Workflow since Account is closed."
  End If

End Function
```

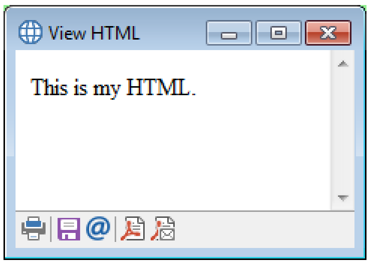
Depending on the value of the `text` parameter returned from the event, the following will occur.

- Blank
  - Nothing will happen.

- Starts with one of the following prefixes, separated by a pipe character:
  - **HTML**
    - E.g.

      ```
      HTML|This is my HTML.
      ```

    - An HTML viewer will be displayed exactly the same as when using an HtmlShowScriptEvent Application Shortcut:

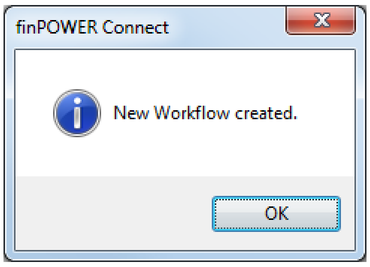


This is my HTML.

  - **INFORMATION** or **INFO**
    - E.g.

      ```
      INFORMATION|New Workflow created.
      ```

    - A message box displaying an information icon:

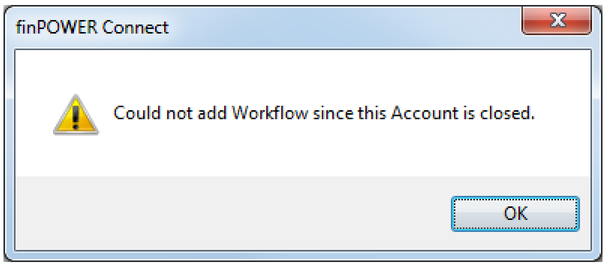


New Workflow created.

  - **WARNING**
    - E.g.

      ```
      WARNING|Could not add Workflow since this Account is closed.
      ```

    - A message box displaying an alert icon:

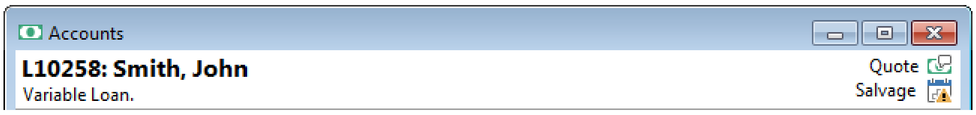


Could not add Workflow since this Account is closed.

Form Summary Text

As of finPOWER Connect 2.02.05, the form heading area can be formatted as HTML.

The following shows the built-in form heading for the Accounts form:

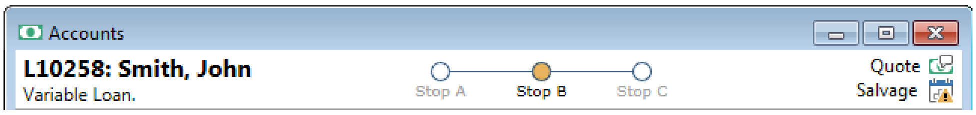


Accounts

L10258: Smith, John

Variable Loan.

And this is a custom form heading using HTML:



Accounts

L10258: Smith, John

Variable Loan.

A form's heading is customised using a 'Summary Text' type Script. This section deals only with customising the form heading to display HTML.

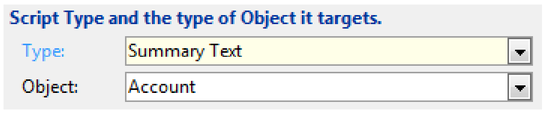
The following sample code can be used to produce the custom HTML form heading (for Accounts of type 'VL') shown above:

```
Option Explicit On

Public Function Main(source As Object,
                     target As iseSummaryTextTarget,
                     contextData1 As Object,
                     contextData2 As Object,
                     ByRef text As String) As Boolean

  Dim Account As finAccount
  Dim sb As finHtmlSummaryPage2StringBuilder
  Dim SummaryText As ISSummaryText
  Dim WorkflowTrainStops As finWorkflowTrainStops

  ' Assume Success
  Main = True

  ' Initialise
  Account = DirectCast(source, finAccount)
  SummaryText = finBL.CreateSummaryText()

  ' HTML Form Heading
  If Account.AccountTypeId = "VL" Then
    ' Respect Background Colour (button strips need this to display correctly)
    SummaryText.BackgroundColour = finBL.AccountFunctions.GetWorstClientStatusColourHtml(Account,
                                                                                           False)
    If Len(SummaryText.BackgroundColour) = 0 Then SummaryText.BackgroundColour = "white"

    ' Build HTML
    sb = finBL.CreateHtmlSummaryPage2StringBuilder()
    With sb
      ' Initialise (set page background colour to match that assigned above)
      .Initialise(iseSummaryPageTarget.Form)
      .HtmlHeaderBegin("", isefinSummaryPageStyle.FormHeading)
      .HtmlHeaderEnd("", "background-color:" & SummaryText.BackgroundColour)

      ' Begin layout table
      sb.Append("<table>")
      sb.Append("<tr>")

      ' Heading (use helper function to match the form fonts etc)
      sb.Append("<td>")
      sb.AppendFormHeading(finBL.HtmlEncode(Account.AccountId & ": " & Account.Name),
                           finBL.HtmlEncode(Account.GetSummaryText(False, False)),
                           "240px")
      sb.Append("</td>")

      ' Train Stops diagram
      WorkflowTrainStops = finBL.CreateWorkflowTrainStops()
      WorkflowTrainStops.BulletShape = isefinWorkflowTrainStopBulletShape.Circle
      With WorkflowTrainStops
        .AddCustom("Stop A", False)
        .AddCustom("Stop B", True)
        .AddCustom("Stop C", False, "app://FormShow?form=Workflows&id=1234")
      End With

      sb.Append("<td>")
      .Append(finBL.SummaryPageStandardBlocks(False).Workflow_TrainStops(Nothing,
                                                                  WorkflowTrainStops))
      sb.Append("</td>")

      ' End layout table
      sb.Append("</tr>")
      sb.Append("</table>")

      ' Finalise
      .HtmlFinalise()
    End With

    ' Set Summary Text HTML
    SummaryText.Html = sb.ToString(True)

    ' Must return Summary Text object as XML (for backward compatibility)
    text = SummaryText.ToXmlString()
  End If

End Function
```

The following are important points of note in the above example:

- The `SummaryText.BackgroundColour` is set for both the `ISSummaryText` object and as the background colour of the HTML page.
- A `finHtmlSummaryPage2StringBuilder` object is used and:
  - The style is initialised to 'FormHeading' to ensure the internal style sheet is picked up.
  - The `AppendFormHeading` method is used to create the heading and summary text so that they appear similar to the form heading in the finPOWER Connect User Interface.
    - This method can also specify `width`, `maxWidth` and `minWidth`.
      - These are all specified as HTML values, e.g., 400px or 50%

The Script should be configured as follows:



Script Type and the type of Object it targets.

Type:	Summary Text
Object:	Account

NOTE: When using a customised HTML form heading, always test on both a normal and high-DPI machine.

WARNING: Use this functionality with caution since there is no guarantee that future versions of finPOWER Connect will keep the same style of size of form heading.

Appendix A – Guidelines

Appendix B – Icon Overlays

HTML Icon templates and also Page Set Icon type Grid columns can specify an overlay to add to the icon. The following table lists some of the more common overlays:

Overlay	Name
➕	Add
✖	Cancel
✓	Check
▰	Clear
✏	Edit
❗	Error
🔭	Find
🌐	Internet
💾	Save
🔍	Search
⚠	Warning

    - Columns
    - Blocks
    - Other?
    - Raw HTML
    - Targets, e.g., Form, Report, Web etc
- ScriptInfo
    - Always use ScriptInfo.ScriptId rather than hard-coding Script Ids
- Overriding existing built-in blocks
    - Editing
    - Completely overriding
- Summary Tables
    - Simple Wiki reference
    - Info table vs Data table
- Application Shortcuts
    - List of ALL Actions and parameters
    - Including how to use with Summary Pages
        - HTML and URL encoding
            - Using ISApplicationShortcut object instead of forming URL String.
    - J013678
        - This job is to create method to create different types of application shortcuts, e.g., CreateApplicationShortcutFormShow("clients", "c10000")
    - Limitations of FormActions, e.g., Summary Page has to be running from the Accounts form to perform an action to show the suspension wizard etc.
- Events (these are really Application Shortcuts and should therefore come after/ before?).
    - Showing HTML
    - Performing action and showing message etc
- Structure
    - E.g., GetPage() function
    - Constants
- Standard Blocks
- Style guidelines
- Samples
    - Mainly point to built-in pages and explain how to expand etc.