insightsoftware

Designer Guide

Longview

Version 26

Document Information

Notices

Copyright

The information contained in this document represents the current view of insightsoftware.com on the issues discussed as of the date of publication. This document is for informational purposes only. insightsoftware.com makes no representation, guarantee or warranty, expressed or implied, that the content of this document is accurate, complete or up to date.

Disclaimer

This guide is designed to help you to use the Longview applications effectively and efficiently. All data shown in graphics are provided as examples only. The example companies and calculations herein are fictitious. No association with any real company or organization is intended or should be inferred.

Contents

Introduction to Longview

Longview provides corporate performance management (CPM) software that leading companies use to drive performance with speed, visibility, and financial integrity. Since 1994, many of the world's most respected companies have been using our technology platform to create a single repository of financial truth from which statutory consolidation, management reporting, financial planning, modeling, analysis, budgeting, forecasting, and strategic tax can be performed quickly and accurately, enterprise wide.

Longview enables enterprise clients to collect, store, analyze, and report on data in real-time by automating, centralizing, and standardizing any one or combination of the following key financial processes: Planning, Budgeting, Forecasting, Consolidation, Financial Close Management, Profitability Analytics, Statutory, XBRL Financial Reporting and Tax Provisioning. With Longview customers can reduce overreliance on spreadsheets, improve transparency, and regain control of these key finance functions.

Longview Tax calculates your company's global tax charge, effective tax rate, and deferred taxes for tax provisioning purposes. Since Longview Tax uses the same technological platform as your corporate performance management solution, the tax reporting process is directly integrated into the corporate close process. As one solution, consolidated pre-tax income can be reported by legal entity to accurately calculate consolidated income tax charges and deferred taxes.

For more information on purchasing Longview Tax, contact your Longview Account Manager. Web services are a standardized way of integrating applications over the Internet or Internet protocol-based networks. Web services rely on certain software standards including Extensible Markup Language (XML), Simple Object Access Protocol (SOAP), Web Service Definition Language (WSDL) and Universal Description, Discovery & Integration (UDDI).

About this guide

This guide includes basic information on how to install your Longview system. The following sections indicate conventions that are used in this guide.

Warnings and notes

This guide uses the following conventions for warnings and notes:

Caution: Warnings provide cautionary information on the possible effect of certain actions, including the unintentional deletion of data. Be sure to read and understand all warnings before performing a related procedure.

Note: Notes provide additional information to help you understand your Longview system better. They also provide important information on exceptions to general guidelines.

Procedures

There may be several ways to perform a procedure in your Longview system.

- You may be able to choose a menu command. For example, to open a file, you can choose **Open** from the File menu. In this documentation, we use: Choose **File** > **Open**.

- You may be able to use a keyboard equivalent. For example, to exit you can press the **Alt** key and then the letter **F** to open the File menu, and then press **X** for Exit.
- You may be able to click a button or icon. If a menu command has an equivalent button or icon, an illustration of the button or icon may appear in the margin.

In this documentation, we may not describe all methods to carry out a task. Use whichever method you prefer. Depending on the task you are performing, certain methods may not be available.

Contacting Longview

Questions? We are ready to help. For contact information for Longview, visit our web site at insightsoftware.com/Longview/.

Apps Category

The apps category is divided into folders containing items based on their purpose.

- Data Collection: The data collection folder contains Mappings, and Data Table Input, Data View Input and Data Import apps.
- Export: The export folder contains Mappings, and Data Area Export apps. Data area export apps come in single value and multiple value export flavors.
- Reports: The reports folder contains Attribute-Based Report, Intercompany Report, and Validation report apps.
- Events: The events folder contains General and Calculated Journal Entry events, and Event Trigger apps.
- Configuration: The configuration folder contains Free Form apps and Configuration Libraries.

Navigation pane

If your system administrator has configured your system to use a custom file path for your system specific content, items in the navigation pane will appear in either normal or *italic* text.

If your system administrator has not configured your system to use a custom file path, all items will appear in *italic* text.

If you are using a custom file path

Items displayed in italic text are custom items either created specifically for your system or customized from and out-of-box item.

If you right-click on a custom item, you will have these options:

- Delete: Deletes the custom item from the server.
- Duplicate: Creates a new item with the same configuration as the original item.

If you right-click on a customized out-of-box item, you will have these options:

- Duplicate: Creates a new item with the same configuration as the original item.
- Revert: Deletes the customized copy of the item and reverts to the out-of-box version.

If you right-click on an out-of-box item, you will have these options:

- Duplicate: Creates a new item with the same configuration as the original item.
- This new item will be saved as a custom item.

Administrative Toolbar

Longview Designer includes an administrative toolbar. The buttons that appear on the administrative toolbar may vary depending on the tool or editor you have open in the workspace:

Button	Description
Save	Click the **Save** button to save any changes made in the current view. **Note:** The save button is disabled for an out-of-box app.
Run	Click the **Run** button to run the current app. This button is not available for Data Import Apps, Events and Configuration Libraries.
Customize	This button only appears for an out-of-box item. Click the **Customize** button to create a customized version of the item.
View Out of Box App	This button only appears for customized out-of-box items. Click the **View Out of Box App** button to open the original out-of-box item in a new tab. The out-of-box is read-only.
Refresh	Click the **Refresh** button to update all information in the current view to reflect any changes that any other users have made. This button is only available for Data Import Apps.
Duplicate	Click the **Duplicate** button to create a new item or app with the settings in the current view.
Help	Click the **Help** button to open context-sensitive Help related to the appropriate tool or editor.

Create An App, Event, Or Configuration Library

You can create an app, event, or configuration library by:

- Starting with a template, or
- Duplicating an existing app

To create a new app, event, or configuration library from a template:

1. Navigate to the Design module.
2. Expand the folder that relates to the purpose of the app, event, or configuration library you which to create.
3. Expand the folder of the specific type of app, event, or configuration library.
4. Click **New**. A new tab appears with the properties page open.

To create a new app, event, or configuration library by duplicating an existing one:

1. Navigate to the Design module.
2. Expand the folder that relates to the purpose of the app you which to create.
3. Expand the folder of the specific type of app, event, or configuration library.
4. Expand the Existing folder under the specific type of app, event, or configuration library.

5. Right-click on the **item** you wish to copy and select **Duplicate**. A new tab appears with the properties page open, containing all the settings and documents in the duplicated item.

To customize an out of box app, event, or configuration library

1. Navigate to the Design module.
2. Expand the folder that relates to the purpose of the app you which to create.
3. Expand the folder of the specific type of app, event, or configuration library.
4. Expand the Existing folder under the specific type of app, event, or configuration library.
5. Click on the **item** you wish to customize. The item opens in a tab. You will see a yellow message bar indicating that the item is read-only.
6. Click the **Customize** button in the toolbar and make any changes you require.
7. When finished making changes, click **Save**.

Specifying properties

Properties are general information about the app, event, or configuration library. The name of an app is displayed in the app publisher interface and Longview Client's navigation pane. The name of an event or configuration library is only referenced in configuration and generally should be kept short.

Theme and window size properties are only available for apps.

Property values for name, description, version, template name, and template version can be retrieved during the execution of an app using the GetAppConfig procedure function.

Field	Notes
Name	Type a name for the app. The name must be unique and can have a maximum of 94 characters, including spaces. You cannot include the slash (/), backslash (\ ), colon (:), ampersand (&), asterisk (*), question mark (?), double quotation mark ("), the right-angle bracket (>), left angle bracket (<), pipe (|), or period (.) in the app name.
Description	Type a description for the app. The description can have a maximum of 100 characters, including spaces.
Version	Type a version to display in brackets after the description in the title bar for the data import app. The version can have a maximum of 100 characters, including spaces. This field is optional.
Template version	Type a version to display in brackets after the description in the title bar for the data import app. The version can have a maximum of 100 characters, including spaces. This field is optional.

Field	Notes
Theme	Specify the look and feel of the app using one of the following values: ■ Longview — Applies Longview styling to messages and prompts. ■ Windows — Applies Windows styling to messages and prompts. The default is Longview. **Note:** Theme only applies when the app is launched outside of Longview Client.
Window size	Select the window size of the data import app using one of the following options: ■ Maximized — Specifies that the application window opens maximized. ■ Specify in Pixels — Lets you specify the window width and height in pixels. The default size in pixels is 1024 by 768. ■ Specify as Percentages — Lets you specify the window width and height as percentages. The default size in percentages is 100 by 100. The default window size is Maximized. **Note:** Window Size only applies when the app is launched outside of Longview Client.

Setting troubleshooting options

Troubleshooting settings are used to set optional output to help find and correct issues related to apps or events. Troubleshooting options are not available for configuration libraries, but the troubleshooting options of the app or event that uses the configuration library are applied.

Caution: Some troubleshooting options a have performance impact and it is important to only enable troubleshooting when absolutely required. Be sure apps are deployed to production systems with troubleshooting mode disabled.

Field	Notes
Enable troubleshooting mode	Check this to enable troubleshooting mode. When disabled no troubleshooting occurs, so options can be left checked for future use. When enabled any error messages provide full detail, instead of just the description of the error.

Field	Notes
Re-use file for log and history	Check this to re-use the same history and log file name on subsequent execution of an app. This results in all files generated during execution to be written to the log path without the date-time sub-folder. The content of each file will only hold the content of the last execution of the app. This aids in using monitoring tools on files during development troubleshooting. **Note:** This setting applies to only Apps and does not impact events, even when selected.
Enable troubleshooting messaging	Check this to enable troubleshooting specific messages to be displayed during execution. Trouble shooting messages must be configured in the app or event.
Enable performance profiling	Check this to have a report of the execution performance output at the end of execution. You can use this output to identify where performance issues are occurring. For detailed information on the output, see "the Profile command in the *Longview Developer's Guide*".
Enable detailed history logging	Check this to create a log containing all the commands executed. **Caution:** Detailed history logging has a large performance impact and should be used as required, and never enabled in a production system.
Enable history timer	When detailed history logging is enabled, check this to output the execution time of each command recorded in the log.

Delete An App, Event, Or Configuration Library

If you have the appropriate authorization, you can use Longview Client to delete an app. You cannot delete an app that is still published to at least one group. Before you delete an app, unpublish it from all groups.

For more information, see Unpublishing Apps From User Groups.

Input Tools

Input app templates provide the ability to add input tools to the view allowing the user to quickly enter values.

Spread behavior

Quick Input (spread evenly) and Quick Spread have certain behaviors to be aware of. Each calculates a residual value based on the unlocked cells in the target area, spreading the remainder to the lock cells using the source pattern. This is commonly used in forecast updates as some of the periods contain actual data and should not be updated. To configure an input app correctly in this scenario, configure the app as follows:

1. In the lock spec, exclude the symbols that should not be updated
    - In a forecast this would be the actual periods. The set of forecast periods can be retrieved from the system attribute SGPForecastForecastPeriods.
2. In the query spec, include all the symbols to be displayed.
3. In the view, display all symbols from the query spec.
    - The symbols excluded form the lock will be protected from update by the user.
    - Spread allocations will not affect these cells.
    - The allocated amount will be adjusted to spread the difference between the amount entered and the amount contained in the protected cells.

If the user attempts a spread on a target area it will be allowed, but no data will be changed. Information can be provided to the user that the target area is read-only by providing a list of un-locked symbols. To provide a list of unlocked symbols, do the following:

1. For the inner column dimension, set the variable VIEW_ReadOnlyColumnList to contain a list of the names of the unlocked symbols.
2. For the inner row dimension, set the variable VIEW_ReadOnlyRowList to contain a list of the names of the unlocked symbols.

Note: The list of read-only symbols must contain the individual leaf symbol names, and not any hierarchy specifications.

Copy across

The copy across tool allows the user to copy the value in a cell across to all related cells. This tool is available on leaf cells and copies values to all leaf cells to the right, under the same root symbol, in the data area. Any cells that are indicated as read-only are skipped.

Adjust values

The adjust values tool allows the user to adjust values starting with the selected cell by a percentage or an absolute amount. This tool is available on leaf cells and adjusts all leaf cells to the right, under the same root symbol, in the data area. Any cells that are indicated as read-only are skipped.

Pattern spread

The pattern spread tool allows you to quickly populate the leaf cells related to a selected pattern. Pattern spread supports using pre-existing patterns set up by the System Administrator.

For more information on creating patterns, see “Managing Patterns” in the *Administrator Guide*.

This tool is only available in a parent cell and affects all leaf cells related to the parent.

Quick input

The quick input tool allows the user to copy a value or spread a value evenly across several related cells. This tool is only available on a parent cell and affects all leaf cells related to the parent cell.

Quick spread

The quick spread tool allows the user to spread a value across several related cells based on a pattern in another row. This tool is only available in a parent cell and affects all leaf cells related to the parent. The value to be spread can be manually entered or can be derived from the pattern source value and adjusted by a percentage and/or absolute amount.

Import

The import tool allows the user to import data from a file into the underlying data area. The data imported will replace the existing values for the intersections specified. If the user chooses to clear the data prior to import, all other cells will be set to 0. If the user does not choose to clear the data prior to import all cells not specified in the file will retain their current value.

File format:

```
Symbol<delimiter>...Symbol<delimiter>Value
```

delimiter can be:

- ,
- ;
- {

Example:

```
ACCOUNT,TIMEPERIOD,ENTITY,CURRENCY,1000
```

Specify the data area definition

Use Data Area Definition to specify the data area to be queried for the export. Open the data area definition to specify its properties.

Note: The name of the data area definition is Main.lvdsp and the name of the data area created using it is daMain.

1. For the Name, enter the name of your Trigger Area.

   Note: The name must 29 or fewer characters and only contain alphanumeric characters and underscore “_”.

2. In the Options table, complete the following fields:

    a. Data type: Specify the data to be queried using one of the following options:

    - ADJUSTED — Indicates that data queries include adjustments made via journal entries.
    - UNADJUSTED — Indicates that data queries exclude adjustments made via journal entries.

    Default is ADJUSTED.

    b. Download type: Specify the type of data to be downloaded using one of the following options (download option specific in brackets):

    - Exclude parent values — Indicates that parent data is excluded in the download (STANDARDLEAFONLY)
    - Exclude parent and calculated values — Indicates that only leaf data that was submitted to the database via import, input or event calculation is included in the download (LEAFDATA).

    Default is All data.

3. In the Trigger Area table, complete the following fields for each dimension:

> **Note:** The Spec and Level fields are not applicable if a symbol contains a variable.

    a. Symbols: Type a symbol name. Alternatively, you can use the symbol selector. To add more than one symbol for a dimension, select a **row** containing the dimension for which you want to add another symbol and click **Symbol**. To move a symbol up or down within a dimension, click **Move Up** or **Move Down**.

> **Note:** To use the symbol selected by the user or defined by an attribute, enter the name of the dimension as a variable, for example $ENTITIES$

    b. Spec: Indicate the symbol specification using one of the following values:

    - All: Includes all symbols within the selected symbol's hierarchy.
        - Leaf: Includes only leaf symbols within the selected symbol's hierarchy.
        - Parent: Includes only parent symbols within the selected symbol's hierarchy.
        - Root and Parent: Includes only root and parent symbols within the selected symbol's hierarchy.

> **Note:** To use the symbol selected by the user or defined by an attribute, enter the name of the dimension as a variable, for example
> $ENTITIES$

c. Level: Specify the number of levels down from the symbol to be included in the data area.

4. Use Additional Configuration to specify any other Data Area Definition features you wish to use. Additional Configuration is a code editor that allows you to enter data area definition functions to be included in the data area.

   For more information on the code editor, see Using the code editor.

   Typical use of Additional Configuration in a data area definition is to

   - Add an attribute filter using the AttributeFilter function
   - Add a schedule to the using the Schedule function
   - Add a temporary symbol using the TempSym function

Specifying forms

Use Forms to capture user input:

- In an action within an input template
- During post-selection in an app

For each form, complete the following fields:

- Name: Enter the name of the form.
- Title: Enter the title for the form. This is optional. If left blank the title of the form will be the same as the description of the app.
- Width (in pixels): Enter the width of the form. If not specified, the default width of 450 is used.
- Height (in pixels): Enter the height of the form. If not specified, the height is determined by the number of controls in the form.
- Format for the date value returned: Enter the format a date is stored in an application framework variable when returned from a form. The format in the form is determined by the user's regional settings.
- Form controls: Add the controls to be displayed in the form via the code editor. For more information on the code editor, see Using the code editor.

Specifying models

Use Models to create models that are executed as part of the app, event, or configuration library.

To create a model:

1. Right-click on **Models** and select **New Document**.
2. For the Name, enter the name of your Model.
3. In the code editor, enter the code required for the model calculation. For more information on models, see Developing Longview Models in the *Longview Developer's Guide*.

Specifying procedures

Use Procedures to create any additional procedures that are required as part of the app, event, or configuration library.

To create a procedure:

1. Right-click on **Procedures** and select **New Document**.
2. For the Name, enter the name of your Procedure.
3. In the code editor, enter the commands to run when the procedure is executed. For more information on procedures, see Developing Longview Procedures in the *Longview Developer's Guide*.

Specifying Symbol Maps

Symbol Maps(.lvmap) specify instructions when the symbol names in your data target do not match the symbol names in your Longview database.

For more information, see Developing Longview ImportSpecs, ExportSpecs, and external Maps in the *Longview Developer's Guide*.

For more information on using the code editor, see Using the code editor.

Specifying symbol selections

Specifying pre-selection commands

Use pre-selection to perform any commands before symbol selection. The typical use of pre-selection is to perform advanced logic to determine default selections or set the range of possible symbols. Pre-selection commands are executed after symbol selection settings are applied from the app configuration, so all symbol selection variables have been created at this point.

See Generating a symbol selection form for details on symbol selection variables.

Note: Using pre-select commands to set default or symbols overrides the settings in User Selections.

Variable	Notes
<Dimension>Default	Specify the default selection in the symbol selector, for the Dimension.
<Dimension>Symbols	Specify the list of available selections in the symbol selector for the Dimension.

Example: Setting the list of possible symbols to all YTD periods

```
Create VARIABLE ytdPeriodList AS RANGE
Set VARIABLE ytdPeriodList = "[[SYSTEM,SGPTimePeriodsYTD,DBDEFAULT]]"
For EACH ytdPeriod in ytdPeriodList
    Set VARIABLE TIMEPERIODSSymbols = ListAppend(TIMEPERIODSSymbols,
"$ytdPeriod$#99")
Next
```

Specifying user selections

Use user selections to specify dimensions for which the user will be prompted to choose symbol(s). For each dimension that the user will choose symbols, right-click on the User Selections folder and choose Add Dimension. The dimension settings page will appear.

Variable	Notes
Dimension	Select the dimension the user will be prompted to select symbol(s) for.
Available selections	Select a symbol to limit the possible symbols available to the user. If left blank the user will be able to select from all symbols available in the dimension. The available symbols will always be limited by user access.
Default selection	Select a symbol to specify as the default selection. This can either be a database symbol, an attribute symbol, or a floating time period. **Note:** Attribute symbols and floating time periods will only be available if they are defined in your system. For more information, see “Specifying attributes in Longview Application Administrator” in the *Longview Analysis and Reporting Guide*.
Selection required	Check this to force the user to select a symbol to continue.
Allow leaf selection	Check this to allow the user to select a leaf symbol.
Allow parent selection	Check this to allow the user to select a parent symbol.
Allow read-only selection	Check this to allow the user to select a read-only symbol.
Allow multiple selections	Check this to allow the user to select more than one symbol.
Attribute filter	Enter an attribute filter expression to further limit the possible symbol selections. For more information see SymbolSelector in the *Longview Developer’s Guide*.

Specifying attribute-based selections

Use Attribute Based Selections to specify selections for dimensions that are deter- mined by an attribute value. For each dimension to be determined via an attribute, right-click on Attribute Based Selections and select Add Dimension.

Variable	Notes
Dimension	Select the dimension to select symbol(s) using an attribute value.
Attribute class	Select the class of attribute to use.
Attribute name	Select the attribute.
Source dimension (SYMBOL class)	If the attribute class is SYMBOL, select the source dimension. **Note:** The source dimension must be a dimension specified as a user selection or attribute-based selection.

Note: The variable created to hold the attribute-based symbol will have the same name as the dimension. The variable created will be a STRING type.

Example, to set the currency selection to be the functional currency of the selected entity:

1. Select the **CURRENCIES** dimension.
2. Set the attribute class to **SYMBOL**.
3. Enter “ZGPNativeCurrency” for the attribute name.
4. Select ENTITIES for the source dimension.

Example, to set the time period selection to be the current period:

1. Select the **TIMEPERIODS** dimension.
2. Set the attribute class to **SYSTEM**.
3. Enter “SGPCurrentPeriod” for the attribute name.

Specifying post-selection commands

Use post-selection to perform any commands after symbol selection, but before any data areas are created. The typical use of post-selection commands is to perform advanced symbol selection tasks. For example, you might use post-selection to create a list of all standard intercompany accounts. In addition, you may want to use a form to further refine selection, or instead of the template provided symbol selection.

Caution: Post-selection commands are executed as part of symbol selection within a view. Any Create VARIABLE commands will cause an error during symbol reselection. If “Allow user to change selections in view” is allowed, either create variables only in pre-selection, or wrap variable creation in VariableExists checks.

Example: Get a list of all standard intercompany accounts

```
If Not(VariableExists("ACCOUNTS"))
    Create VARIABLE ACCOUNTS AS RANGE
END If
Set VARIABLE ACCOUNTS = CreateList(SYMBOLS, DATABASE, ACCOUNTS, "TB###")
Set VARIABLE ACCOUNTS = FilterList(ACCOUNTS, "
[[SYMBOL,ZElimICSchTransactions,THIS]] == 'ICStandard'")
```

Example: Specifying multiple attribute time periods (current forecast and budget period)

```
If Not(VariableExists("TIMEPERIODS"))
    Create VARIABLE TIMEPERIODS AS RANGE
END If
Set VARIABLE TIMEPERIODS = "[[SYSTEM,SGPForecastPeriod,DBDEFAULT]]" Set
VARIABLE TIMEPERIODS = ListAppend(TIMEPERIODS, "
[[SYSTEM,SGPBudgetPeriod,DBDEFAULT]]")
```

Example: Specify temporary symbols mixed with real symbols

```
If Not(VariableExists("TIMEPERIODS"))
    Create VARIABLE TIMEPERIODS AS RANGE
END If
Set VARIABLE TIMEPERIODS = "[[SYSTEM,SGPForecastPeriod,DBDEFAULT]]"
Set VARIABLE TIMEPERIODS = ListAppend(TIMEPERIODS, DIFF")
Set VARIABLE TIMEPERIODS = ListAppend(TIMEPERIODS, "
[[SYSTEM,SGPBudgetPeriod,DBDEFAULT]]")
```

To capture additional user input via a form, it is required that the button clicked be copied to the FORM_ ButtonClicked variable to ensure expected app execution. Specifically, app execution continues if FORM_ButtonClicked has a value of “OK”.

Example: Capture additional user input via a form

```
Show FORM USING Custom.lvfrm
Set VARIABLE FORM_ButtonClicked = $LVS_BUTTONCLICKED$
```

Allowing currency re-selection in a data view input app

There may be cases where the initial selection in the app is based on the functional currency of the selected entity, but you want the user to be able to change the currency.

To accomplish this, follow these steps:

Note: Substitute CURRENCY for the name of your currency dimension, or use an attribute token or $CORE.DimensionInfo.KeyDimensionList[4]$ for CURRENCY for flexibility.

1. In the pre-selection add the following to support currency selection:

```
//Configure currency selections for re-select in view
Create GLOBALVARIABLE CURRENCYDefault AS STRING
Create GLOBALVARIABLE CURRENCYOptions AS STRING
Create GLOBALVARIABLE CURRENCYSymbols AS RANGE
Create GLOBALVARIABLE oldSelectDimensionList[] AS STRING //used to
restore symbol selection to original options

Set VARIABLE CURRENCYOptions = "11000"
Set VARIABLE CURRENCYSymbols = "SOURCEC###"

Create VARIABLE isCurrencySelection AS NUM //used to track in post-
selection if the is changing currency only.
```

2. In the Post-Selection add logic to set the native currency on entity selection:

```
If Not(VariableExists("CURRENCY"))
    Create VARIABLE CURRENCY AS STRING
END If

If $isCurrencySelection$ //reset selection options to allow regular
symbol selections
    Set VARIABLE FORM_SelectDimensionList = ListAppend(CORE_EmptyList,
oldSelectDimensionList)
    Set VARIABLE isCurrencySelection = 0
Else
    Set VARIABLE CURRENCY = "[[SYMBOL,ZGPNativeCurrency,$ENTITIES$]]"
//apply currency of selected entity
END If
```

3. Add a View Action to allow the user to change currency (this will be a new button on the toolbar).

```
Set VARIABLE oldSelectDimensionList = ListAppend(CORE_EmptyList, FORM_
SelectDimensionList)
Set VARIABLE isCurrencySelection = 1
```

```
Set VARIABLE FORM_SelectDimensionList = ListAppend(CORE_EmptyList,
"CURRENCY")


Run PROCEDURE "VIEW\OnReselect.lvpro"
```

Specifying symbol selections no pre-Post

Specifying user selections

Use user selections to specify dimensions for which the user will be prompted to choose symbol(s). For each dimension that the user will choose symbols, right-click on the User Selections folder and choose Add Dimension. The dimension settings page will appear.

Variable	Notes
Dimension	Select the dimension the user will be prompted to select symbol(s) for.
Available selections	Select a symbol to limit the possible symbols available to the user. If left blank the user will be able to select from all symbols available in the dimension. The available symbols will always be limited by user access.
Default selection	Select a symbol to specify as the default selection. This can either be a database symbol, an attribute symbol, or a floating time period. **Note:** Attribute symbols and floating time periods will only be available if they are defined in your system. For more information, see “Specifying attributes in Longview Application Administrator” in the *Longview Analysis and Reporting Guide*.
Selection required	Check this to force the user to select a symbol to continue.
Allow leaf selection	Check this to allow the user to select a leaf symbol.
Allow parent selection	Check this to allow the user to select a parent symbol.
Allow read-only selection	Check this to allow the user to select a read-only symbol.
Allow multiple selections	Check this to allow the user to select more than one symbol.
Attribute filter	Enter an attribute filter expression to further limit the possible symbol selections. For more information see SymbolSelector in the *Longview Developer’s Guide*.

Specifying attribute-based selections

Use Attribute Based Selections to specify selections for dimensions that are deter- mined by an attribute value. For each dimension to be determined via an attribute, right-click on Attribute Based Selections and select Add Dimension.

Variable	Notes
Dimension	Select the dimension to select symbol(s) using an attribute value.
Attribute class	Select the class of attribute to use.
Attribute name	Select the attribute.
Source dimension (SYMBOL class)	If the attribute class is SYMBOL, select the source dimension. **Note:** The source dimension must be a dimension specified as a user selection or attribute-based selection.

Note: The variable created to hold the attribute-based symbol will have the same name as the dimension. The variable created will be a STRING type.

Example, to set the currency selection to be the functional currency of the selected entity:

1. Select the **CURRENCIES** dimension.
2. Set the attribute class to **SYMBOL**.
3. Enter “ZGPNativeCurrency” for the attribute name.
4. Select ENTITIES for the source dimension.

Example, to set the time period selection to be the current period:

1. Select the **TIMEPERIODS** dimension.
2. Set the attribute class to **SYSTEM**.
3. Enter “SGPCurrentPeriod” for the attribute name.

Specifying table definitions

This topic details the process for specifying table definitions.

To create a table definition:

1. Right-click on Table Definition and select **New Document**.
2. For the Name, enter the name of your Table Definition.
3. In the columns section the table and columns are defined.
   a. For the table name, select the **table** to use. The list includes all app tables defined in the system.
   b. If you pick a table based on a view it will display an information icon indicating that the table will be read-only.
   c. After selecting a table the columns in the table will be displayed. All columns will be selected by default. Also the columns that define the primary key will be indicated with a key icon. If some or all of the primary key columns are not included, the table will be read only.
   d. You may add or remove temporary columns using the **Add Column** and **Remove Column** buttons.

> **Note:** Temporary columns are added to the table for use with an app, but are not submitted to the database.

e. For each column, complete the following fields:

- Include: Check each **column** to be included in the query. Use the check box in the header to select all columns.
- Column: For persistent table columns, displays the name of the column. For temporary columns, enter the name of the column.
- Nullable: Indicates whether the column can be saved with no value specified. If any non-nullable columns are excluded the table will be read-only.
- Type: For persistent table columns, displays the data type of the column. For temporary columns, select the **data type** for the column. If you select Symbol[Dimension], you will be prompted to select the dimension that applies. If you select Symbol [DimensionColumnName], you will be prompted to select a column (of type Dimension).
- Values (String): For String columns allows you to optionally define a list of values the user can select from. Enter a | delimited list of values. Optionally, enter a | delimited list of display values by placing a semi-colon after the list of values.

  > **Note:** You can also use variables to define the list of values and display values.

- Values (Symbol): For Symbol columns allows you to limit the symbols available. Enter a | delimited list of symbol specifications. You can also use a form to select the symbol specifications by clicking the **search** icon.

  > **Note:** You can also use a variable to define the list of symbol specifications.

- Values (User): For User columns, allows you to limit the users available for selection. Enter a | delimited list of symbol specifications. You can also use a form to select the symbol specifications by clicking the **search** icon.

  > **Note:** You can also use a variable to define the list of users and groups.

- Values (User List): For UserList columns, allows you to limit the users available for selection. Enter (or select) a | delimited list of user and group names.

  > **Note:** You can also use a variable to define the list of users and groups.

- Allow Override: For String columns with Values defined, indicates whether the user can enter a value that is not in the list of values.
- Min/Max (Date): For date columns, allows you to set a valid range of selectable dates.

  Note: The interface only supports explicit dates. To specify the valid date range using variables use the Additional Configuration section to define the date range using the SetColumn function.

- Default (Boolean): For Boolean columns, allows you to indicate if the default value for the column is checked.
- Default (Dimension): For Dimension columns, allows you to set he default dimension to be selected when a new row is added. Default is no default selection.

Working with variables

You can use Longview Designer to create, modify, reorder, or delete variables for your app. Variables are optional. You might include variables in your app if you want to, for example, use an attribute such as the current time period, or if you want to prompt a user to select a symbol or a file. Once you have created variables, you can use them elsewhere in your app by enclosing them in dollar signs ($ $).

For more information, see “Variables” in the *Longview Developer’s Guide*.

You can also use external variables not defined on the Variables page of your app by enclosing them in dollar signs ($ $).

Variables tasks include:

- Creating variables
- Modifying variables
- Reordering variables
- Deleting variables

Creating variables

You can use Longview Designer to create variables for your app. Each variable you specify is created and set to the specified value when your app runs.

To create a variable:

1. Open the Variables section.
2. Do one of the following:
   - To create a new variable at the bottom of the list, click Variable. The new variable appears at the bottom of the variables list.

- To create a new variable below a specific variable, select a row and click Variable. The new variable appears below the selected row.

3. Complete the following fields:

   a. Name: Type the name of the variable. Do not prefix the variable name with “LV_” or “LVS_”.

   b. Type: Specify the type of data to be stored in the variable using one of the following options:

      - String — Designates a string value.
      - Range — Designates a list of symbols.
      - Number — Designates a numeric value.
      - List — Designates a list of string values.
      - NumberList — Designates a list of numeric values.

      The default is String.

   c. Expression: Specify the value that the variable is set to. Based on the Type field this can be a number, a string, or a list of symbols, respectively.

Note: You can include attribute values in an expression. To include an attribute value in an expression, you must first set a variable to the value of the attribute using an attribute token, and then use the variable. For example, you could create a string variable named “currentPeriod” and use the expression “[[SYSTEM,SGPCurrentPeriod,DBDefault]]”. For more information on setting variables, see “Set Variable” in the *Longview Developer’s Guide*. For more information on using attributes, see “Attribute tokens” in the *Longview Developer’s Guide*.

Modifying variables

You can use Longview Designer to modify existing variables for your app.

To modify a variable:

1. Open the Variables section.
2. Select the **variable** you want to modify and make the necessary changes.

Reordering variables

You can use Longview Designer to change the order of variables in your app. The order of variables is important because you cannot use a variable in an expression until you first define the variable. Variables are created in the same order in which they are defined.

To reorder a variable:

1. Open the Variables section.
2. Select the **variable** you want to reorder, and then click either **Move Up** or **Move Down**. The variable moves one row in the indicated direction.
3. If you want to continue to move the selected variable, repeat step 2 until the variable is in the desired location.

Deleting variables

You can use Longview Designer to delete variables from your app.

To delete a variable:

1. Open the Variables section.
2. In the variables list, select the **variable** you want to delete, and click **Delete**. The variable is deleted.

Creating Data Table Input Apps

You can use Longview Designer to create data table input apps.

Specifying settings

You can use Longview Designer to specify global settings for a data table input app.

Field	Notes
Layout type	Use this to specify the layout type of the view. Select Tabbed for a multi-tabbed view layout. Select Single for a single view layout. **Note:** ■ If Single layout type is selected, only one Table View can be defined. ■ If Tabbed layout type is selected, at least one Tab must be created. Tabs are not required for a Single layout type.
Use selection form	Check this if a form will be used for user selections prior to displaying the view. If a form is to be used Selection must be filled out
Use validations	Select Yes to validate the data entered prior to it being submitted. **Note:** This feature requires validations to be configured for the app.
Default export file name	Use this to specify a default name for the target file for export. This is optional and need only be set if you want a default name and export is supported on one or more tables.

Specifying pre-selection commands

Use pre-selection to perform any commands before the selection form is displayed. The typical use of pre-selection is to perform advanced logic to determine default selections or other configuration related to the selection form.

Specifying user selections

Selection is used to capture any input from the user prior to retrieving data and displaying the view. This could be used for flittering options related to the data or view formatting options. Selection is a form and supports all keywords related to the form document.

For more information on designing forms, see the *Longview Developer's Guide*.

Specifying post-selection commands

Use post-selection to perform any commands after the selection form is displayed. The typical use of post-selection is to determine any additional settings that are calculated based on the user's selections.

Specifying the data table definition

Use Data Table Definitions to specify the data tables to be used in the data input view. For each data table definition to be created, right-click on Data Table Definitions and select New Document.

Fill in the data table definition sections:

1. For the Name, enter the name of your Data Table

   Note: The name must 29 or fewer characters and only contain alphanumeric characters and underscore "_". The actual name of the data table in application framework will be "dt" + Name, so if you need to refer to the Main data table in application framework, use dtMain as the name.

2. In the Options table, complete the following fields

Field	Notes
Create this data table	Select this option if you would like the data table to be created prior to the view. **Note:** In some complex cases you may want to define the data table, but only created when required during, save, validate or some other processing.
Download this data table	Select this option if you would like to download data in this data table. **Note:** If you have more than one data table set to download, the last on in the list will be displayed when the layout type is Single.

Field	Notes
Submit from this data table	Select this option if you would like to submit from this data table to the database once the submit button is clicked within the view. **Note:** You cannot set a data table to submit without enabling the "Download this data table" option as well. At least one data table definition in your data table input app must be set to submit
Import spec to use for importing to this table	If the user will be allowed to import data into this table, select the import spec that will be used to process the import. The selections available are determined by the contents of the Import Specs folder.
Export spec to use for exporting from this table	If the user will be allowed to export data from this table, select the export spec that will be used to process the export. The selections available are determined by the contents of the Export Specs folder.

3. In the columns section the table and columns are defined.

- For the table name, select the table to use. The list includes all app tables defined in the system.
- If you pick a table based on a view it will display an information icon indicating that the table will be read-only.
- After selecting a table, the columns in the table will be displayed. All columns will be selected by default. Also, the columns that define the primary key will be indicated with a key icon. If some or all the primary key columns are not included, the table will be read only.
- You may add or remove temporary columns using the Add Column and Remove Column buttons.

> **Note:** Temporary columns are added to the table for use with an app but are not submitted to the database.

- For each column complete the following fields:

Field	Notes
Include	Check each column to be included in the query. Use the check box in the header to select all columns.
Column	For persistent table columns, displays the name of the column. For temporary columns, enter the name of the column.
Nullable	Indicates whether the column can be saved with no value specified. If any non-nullable columns are excluded the table will be read-only.
Type	For persistent table columns, displays the data type of the column.For temporary columns, select the data type for the column. If you select Symbol[Dimension], you will be prompted to select the dimension that applies. If you select Symbol[DimensionColumnName], you will be prompted to select a column (of type Dimension).
Values (String)	For String columns allows you to optionally define a list of values the user can select from. Enter a | delimited list of values. Optionally enter a | delimited list of display values by placing a semi-colon after the list of values. **Note:** You can also use variables to define the list of values and display values.
Values (Symbol)	For Symbol columns allows you to limit the symbols available. Enter a | delimited list of symbol specifications. You can also use a form to select the symbol specifications by clicking the search icon. **Note:** You can also use a variable to define the list of symbol specifications.
Values (User)	For User columns, allows you to limit the users available for selection. Enter (or select) a | delimited list of user and group names. **Note:** You can also use a variable to define the list of users and groups.
Values (User List)	For UserList columns, allows you to limit the users available for selection. Enter (or select) a | delimited list of user and group names. **Note:** You can also use a variable to define the list of users and groups.
Allow Override	For String columns with Values defined, indicates whether the user can enter a value that is not in the list of values.

Field	Notes
Min / Max (Date)	For date columns, allows you to set a valid range of selectable dates. **Note:** The interface only supports explicit dates. To specify the valid date range using variables use the Additional Configuration section to define the date range using the SetColumn function.
Default (Boolean)	For Boolean columns, allows you to indicate if the default value for the column is checked.
Default (Dimension)	For Dimension columns, allows you to set he default dimension to be selected when a new row is added. Default is no default selection.

4. Use additional configuration to:
   - Define sort order.
   - Define any filters (SymbolFilter, UserFilter, or Where).

Specifying the data table view

Use Table Views to specify the views to be used in the data table input app. For each table view to be created, right-click on Table Views and select New Document.

Fill in the table view sections:

1. For the Columns, select the Data Table Definition that will be used with the view. The columns selected in the Data Table Definition appear in the table. For each column complete the following fields:

Field	Notes
Include	Check each column to be included in the query. Use the check box in the header to select all columns.
Column	Displays the name of the column.
Type	Displays the data type of the column.
Description	Allows you to enter a description for the column to be displayed in the view.
Width	Allows you to specify the width of the column in the view.
Protect	Check to protect the column from input.
Decimals (Number)	Allows you to specify the number of decimals displayed for numeric values.
Enabled Criteria	Allows you to specify criteria which allows the user to modify the value of the column in each row of the table.

2. In the Parameters table, complete the following fields:

Field	Notes
Default number of decimals for numeric values	Specifies the number of decimals to display for numeric numbers (0-9). **Note:** The number of decimals for a specific column can be overridden using the ColumnDecimals function in the View Column Settings section.
Default column width	Specifies the width of each column in the view. **Note:** The width for a specific column can be overridden using the ColumnWidth function in the View Column Settings section.
Apply users' layout preferences	Select Yes to allow the user's modifications to the view to be saved and applied when the app is accessed. Select No to lock the settings of the initial view each time the app is accessed. **Note:** The user can modify the view while using the app, but the next time it is executed the view will revert to the one saved in the app.
Use parentheses for negative numbers	Select Yes to display negative numbers with parentheses. Select No to display negative numbers with a negative sign.
Use thousands separator for numbers	Select Yes to display a locale-specific thousands separator based on the operating system setting. Select No to not display a thousand's separator.
Left title text	Enter text to display in the left title area of the table view. This is optional and can be left blank.

3. In the Table Controls table, complete the following fields:

Field	Notes
Add	Check this to allow the user to add new rows to the table.
Delete	Check this to allow the user to delete rows from the table.
Duplicate	Check this to allow the user to add new rows to the table by duplicating an existing row.
Reset	Check this to allow the user to reset a row's values in the case of invalid input.
ResetAll	Check this to allow the user to reset all row values in the case of invalid input.

4. In the Values pane Options, complete the following fields:

Field	Notes
Enable values pane	Select whether the user will be able to open the values pane within the view. Default is Yes.
Values pane title	Enter text to appear in the title area of the values pane. Default is Row Details
Hide values pane initially	Select whether the values panel in initially hidden, if enabled. Default is Yes.

5. In the Import/Export Options table, complete the following fields:

Field	Notes
Allow use of import	Select Yes to allow the user to import data from a file. The file to import must be a text file containing comma-separated values and a header row. The user will be able to clear all data in the view prior to import.
Allow use of export	Select Yes to allow the user to export data to a file. The file exported will contain a header row and comma-separated values.

6. Use Additional Configuration to further configure the view. Additional Configuration is a code editor that allows you to enter view functions to be included in the configuration of the view.
For more information on the code editor, see Using the code editor.
Typical use of View Column Settings in this app would be to:

   - Sort the rows in the view by one or more columns using the SortOrder function.
   - Adding summary rows to the view using the SummaryRow function.
   - Adding a custom editor for a column using the EditProcedure function.

Specifying view actions

Use View Actions to specify custom toolbar and context menu actions with a view. For each action to be created, right-click on View Actions and select New Document.

For each action complete the following fields:

Field	Notes
Name	Enter the name of the action.
Action location	Select whether the action will appear in the toolbar or on the row context menu.
Views to add action to	Enter text to display for the action. For a toolbar location, the text will appear to the right of the icon, if specified. For a context menu this will be the menu text.

Field	Notes
Action text	Enter text to display for the action. For a toolbar location, the text will appear to the right of the icon, if specified. For a context menu this will be the menu text. **Note:** This is required for a context action, but is only required for a toolbar action, if an icon is not specified.
Action icon	Enter the path to the icon to display for the action. **Note:** This is optional for a context action, but is required for a toolbar if action text is not specified.
Action tooltip text	Enter additional information that will appear when the user hovers the mouse on the action. This is optional.
Row restriction expression	Enter an expression to limit the rows the context menu appears on. This is optional and only applied when the action location is Row. The restriction is based on values in one or more columns of a specific row. For more information on the expression, see Action in the *Longview Developer's Guide*. **Note:** When entering an expression that requires quotes use a single quote (') instead of double quotes ("). Example: `{{Column,CASE}} == \'Happy Case\'`
Action	Enter the commands to execute for this action via the code editor. For more information on the code editor, see Using the code editor.

Specifying dynamic actions

Use Dynamic Actions to specify actions that execute as the data is changed by the user within the view. For each action to be created, right-click on Dynamic Actions and select New Document.

For each action complete the following fields:

Field	Notes
Name	Enter the name of the action.
Type	Select whether the action will execute on a table change (Reset All), a row change (Add, Duplicate, Reset, Delete) or when the value in a cell is modified.
Views to add action to	Select the table views to assign the action to.
Cell restrictions	Enter a pipe "|" delimited list of column names to execute this action against. The action will only execute when the value in one of the specified columns is modified.
Action	Enter the commands to execute for this action via the code editor. For more information on the code editor, see Using the code editor.

Specifying edit actions

Use Edit Actions to create a procedure to invoke a custom editor for a column in a data table. Edit actions are used when the value stored in a data table cell is more complex than a simple input. Example: a cell contains the definition of a cell or data area. It could also be used to provide a form used to calculate a value for input rather than requiring a user to manually enter a value that may be based on a calculation.

Specifying post-refresh commands

Use Post-Refresh to perform any commands after the data tables are downloaded but before any refresh calculations are performed. Use Post-Refresh to perform any processing that requires data to be downloaded into the data tables.

Specifying tabs

Use Tabs to add tabs to the view. Each tab can display a different table view. For each Tab to be created, right-click on Tabs and select Add Tab.

Note: Tabs are only required if using a Tabbed Layout Type. For more information, see Specifying settings.

For each tab complete the following fields:

Field	Notes
Data table	Select the data table definition that will be displayed in this tab.
Table view	Select the table view to be used in this tab.
Tab name	Use this to provide a label to display on the tab
Action to run when switching to this tab	Select the command to execute when this tab is switched to by the user. The selections available are determined by the contents of the Tab Commands folder. This parameter is optional. **Note:** If this is the first tab this command will be executed when the view first appears.
Action to run when switching from this tab	Select the command to execute when this tab is switched from by the user. The selections available are determined by the contents of the Tab Commands folder. This parameter is optional.
Background color	Use this to select the background color for the tab. The default color is transparent.

Specifying a tab action

Use tab commands to define commands and functions to be executed when switching between tabs in a tabbed view.

For more information on the code editor, see Using the code editor.

Specifying an import spec

Use import specs to define the specification for importing data to a specific data table. For each import spec to be created, right-click on Import Specs and select New Document.

Fill in the import spec sections:

1. For the Name, enter the name of your Import Spec.
2. Fill in the Import Settings. Many of the import settings are pre-defined so the main requirement for the import spec is to:
   a. Define the fields in the source file, using Set functions.
   b. Define the relationship between fields in the source file and columns in the data table, using the Define function.
   c. Setting the maximum number of errors allowed during import, using the MaxErrors function.
   d. Defining any filter conditions on the records of the source file, using the Search function.

Specifying an export spec

Use export specs to define the specification for exporting data from a specific data table. For each export spec to be created, right-click on Export Specs and select New Document.

Fill in the export spec sections:

1. For the Name, enter the name of your Export Spec.
2. For Options, fill in the text to create the header row of the target file. The header row allows you to provide context to the values appearing in the file when it is exported.
3. Many of the export settings are pre-defined so the main requirement for the export spec is to:
   a. Define the fields in the target file, using Set functions.
   b. Define the relationship between columns in the data table and fields in the target file, using the Define function.

Specifying calculations

Use Calculations to define calculations to be executed against one or more data tables in the app. A calculation is a series of commands executed to perform calculations for one or more columns in a data table. In general, a calculation will:

1. Create a STRING[] variable to hold each row as it is retrieved.
2. Optionally create one or more NUM variables to hold the index of the columns to be tested. This allows for validations to continue to work even if the order of the columns in the table changes.
3. Use the Open CURSOR command to initiate row processing.
4. Use the Fetch command to retrieve rows from the data table.
5. Use the While command to loop through the rows.

> **Note:** You can add a template row processor including the above functions using the Open_Cursor_Template_RowProcessor syntax template available in the code editor. For more information on syntax templates, see Using syntax templates.

6. Use the SetDataTableCell function to set the value for a specific cell in the current row being processed.

Fill in the row processor sections:

1. For the Name, enter the name of your Row Processor.
2. In the Run Processor table, complete the following fields:

Field	Notes
Run processor when data is refreshed	Check this to cause the row processor to execute when data is downloaded initially and whenever the user clicks Refresh.
Run processor after user imports data	Check this to cause the row processor to execute after the user imports data via the import tool. **Note:** Allow use of import must be enabled.
Run processor before data is submitted	Check this to cause the row processor to execute when the user clicks Submit, but before the data is uploaded.
Run processor after data is submitted	Check this to cause the processor to execute when the user clicks Submit, but after the data is uploaded. This is mostly useful when the processor before submitting changes the data in ways that are confusing to a user remaining in the app, for example clearing data that should not be submitted.

3. Specify the commands and functions to execute in the Processor. Processor is a code editor that allows you to enter procedure commands and functions to execute for row processing. For more information on the code editor, see Using the code editor.

Example:

```
Create VARIABLE colPriceIdx AS NUM
Create VARIABLE colRevenueIdx AS NUM
Create VARIABLE colUnitIdx AS NUM
Create VARIABLE revenue AS NUM
Create VARIABLE row[] AS STRING
Create VARIABLE status AS NUM
Set VARIABLE colPriceIdx = GetDataTableColumnIndex(dtMain, "PRICE")
Set VARIABLE colUnitIdx = GetDataTableColumnIndex(dtMain, "UNITS")
```

```
Open CURSOR cursor SELECT "*" FROM dtMain
Fetch FIRST DATATABLEROW FROM cursor INTO row
While $LVS_FETCHSTATUS$ != -1
    If $LVS_FETCHSTATUS$ == 0
        //Insert data table row processing here
        If "$row[$colPriceIdx$]$" != "NULL" AND "$row[$colUnitIdx$]$" !=
"NULL"
            Set VARIABLE revenue = $row[$colPriceIdx$]$ * $row
[$colUnitIdx$]$
            Set VARIABLE status = SetDataTableCell(dtMain, $LVS_FETCHINDEX$,
$colRevenueIdx$, $revenue$)
        END If
    END If
    Fetch NEXT DATATABLEROW FROM cursor INTO row
END While
Close CURSOR cursor
```

Specifying data validations

Use Validations to specify data validations to perform when the user clicks the submit icon. Validation failures can be specified as errors or warnings. Errors will prevent the user from submitting data, while warnings will inform the user of validation failures, but still allow the user to proceed with submitting data.

Note: For the Data Table Input app data there is a single validation procedure for all table views. Be sure to reference the correct data table name in any validation functions.

In many ways, validations for data table input apps are like calculations. Like calculations, validations will require processing each row to check for any validation errors or warnings.

To perform data table validation:

1. Create a STRING[] variable to hold each row as it is retrieved.
2. Optionally create one or more NUM variables to hold the index of the columns to be tested. This allows for validations to continue to work even if the order of the columns in the table changes.
3. Use the Open CURSOR command to initiate row processing.
4. Use the Fetch command to retrieve rows from the data table.
5. Use the While command to loop through the rows.

Note: You can add a template row processor including the above functions using the Open_Cursor_Template_RowProcessor syntax template available in the code editor. For more information on syntax templates, see Using syntax templates.

6. Repeat for each validation.
    a. Write a test or tests to execute a specific validation.
    b. If a test fails, set the VALD_Result variable and run procedure VALD\AddResult.lvpro to add the validation failure result to the list of validation failures.

Example:

```
Create VARIABLE colNameIdx AS NUM
Create VARIABLE row[] AS STRING
Set VARIABLE colNameIdx = GetDataTableColumnIndex(dtMain, "NAME")
Open CURSOR cursor SELECT "*" FROM dtMain
Fetch FIRST DATATABLEROW FROM cursor INTO row
While $LVS_FETCHSTATUS$ != -1
    If $LVS_FETCHSTATUS$ == 0
        //Insert data table row processing here
        If "$row[$colNameIdx$]$" == "NULL"
            Set VARIABLE VALD_Result = "ERROR|$LVS_FETCHINDEX$|Name must be
specified"
            Run PROCEDURE "VALD\AddResult.lvpro"
        END If
    END If
    Fetch NEXT DATATABLEROW FROM cursor INTO row
END While
Close CURSOR cursor
```

Setting Variable VALD_Result

The VALD_Result variable is a string list variable, set as follows (separate each item with a pipe character “|”):

1. ERROR or WARNING to specify the validation failure level.
2. An identifier for the row, usually the primary key.
3. A description of the issue that caused the test to fail.

Specifying forms

Use Forms to capture user input:

1. In an action within an input template
2. During post-selection in an app

For each form complete the following fields:

Field	Notes
Name	Enter the name of the form.
Title	Enter the title for the form. This is optional. If left blank the title of the form will be the same as the description of the app.
Width (in pixels)	Enter the width of the form. If not specified, the default width of 450 is used.
Height (in pixels)	Enter the height of the form. If not specified, the height is determined by the number of controls in the form.
Format for the date value returned	Enter the format a date is stored in an application framework variable when returned from a form. The format in the form is determined by the user's regional settings.
Form controls	Add the controls to be displayed in the form via the code editor. For more information on the code editor, see Using the code editor.

Creating Data View Input Apps

You can use data view input apps to create single or multi-tabbed data input apps to allow users to enter data as required.

Specifying overall settings

You can specify settings for a data view input app that control the type of data displayed, whether the interface has tabs or not, and some of the options available to the user.

Field	Notes
Layout type	Use this to specify the layout type of the data view. Select Tabbed for a multi-tabbed data view layout. Select Single for a single view layout. **Note:** ■ If Single layout type is selected, only one Data Area View can be defined. ■ If Tabbed layout type is selected, at least one Tab must be created. Tabs are not required for a Single layout type.
Use comments	Select Yes to allow the user to enter comments in the data view.
Use line item detail	Select Yes to allow the user to enter line item detail in the data view. **Note:** This will only be available if line item detail is configured for the time periods in the view.

Field	Notes
Use attachments	Select Yes to allow the user to attach files in the data view. **Note:** This will only be available in the data view if file attachments are enabled in the system.
Use validations	Select Yes to validate the data entered prior to it being submitted. **Note:** This option requires validations to be configured for the app.
Data type	Select Adjusted to query adjusted data (unadjusted + journal entries) or Unadjusted to query data excluding journal entries. **Note:** The option selected will apply to all data areas that are downloaded. This can be overridden per data area if required by using the AdjustedDetail function in the Data Area Definition's "Additional Configuration" section.

Specifying pre-selection commands

Use pre-selection to perform any commands before symbol selection. The typical use of pre-selection is to perform advanced logic to determine default selections or set the range of possible symbols. Pre-selection commands are executed after symbol selection settings are applied from the app configuration, so all symbol selection variables have been created at this point.

See Generating a symbol selection form for details on symbol selection variables.

Note: Using pre-select commands to set default or symbols overrides the settings in User Selections.

Variable	Notes
<Dimension>Default	Specify the default selection in the symbol selector, for the Dimension.
<Dimension>Symbols	Specify the list of available selections in the symbol selector for the Dimension.

Example: Setting the list of possible symbols to all YTD periods

```
Create VARIABLE ytdPeriodList AS RANGE
Set VARIABLE ytdPeriodList = "[[SYSTEM,SGPTimePeriodsYTD,DBDEFAULT]]"
For EACH ytdPeriod in ytdPeriodList
    Set VARIABLE TIMEPERIODSSymbols = ListAppend(TIMEPERIODSSymbols,
"$ytdPeriod$#99")
Next
```

Specifying user selections

Use user selections to specify dimensions for which the user will be prompted to choose symbol(s). For each dimension that the user will choose symbols, right-click on the User Selections folder and choose Add Dimension. The dimension settings page will appear.

Variable	Notes
Dimension	Select the dimension the user will be prompted to select symbol(s) for.
Available selections	Select a symbol to limit the possible symbols available to the user. If left blank the user will be able to select from all symbols available in the dimension. The available symbols will always be limited by user access.
Default selection	Select a symbol to specify as the default selection. This can either be a database symbol, an attribute symbol, or a floating time period. **Note:** Attribute symbols and floating time periods will only be available if they are defined in your system. For more information, see “Specifying attributes in Longview Application Administrator” in the *Longview Analysis and Reporting Guide*.
Selection required	Check this to force the user to select a symbol to continue.
Allow leaf selection	Check this to allow the user to select a leaf symbol.
Allow parent selection	Check this to allow the user to select a parent symbol.
Allow read-only selection	Check this to allow the user to select a read-only symbol.
Allow multiple selections	Check this to allow the user to select more than one symbol.
Attribute filter	Enter an attribute filter expression to further limit the possible symbol selections. For more information see SymbolSelector in the *Longview Developer’s Guide*.

Specifying attribute-based selections

Use Attribute Based Selections to specify selections for dimensions that are deter- mined by an attribute value. For each dimension to be determined via an attribute, right-click on Attribute Based Selections and select Add Dimension.

Variable	Notes
Dimension	Select the dimension to select symbol(s) using an attribute value.
Attribute class	Select the class of attribute to use.
Attribute name	Select the attribute.

Variable	Notes
Source dimension (SYMBOL class)	If the attribute class is SYMBOL, select the source dimension. **Note:** The source dimension must be a dimension specified as a user selection or attribute-based selection.

Note: The variable created to hold the attribute-based symbol will have the same name as the dimension. The variable created will be a STRING type.

Example, to set the currency selection to be the functional currency of the selected entity:

1. Select the **CURRENCIES** dimension.
2. Set the attribute class to **SYMBOL**.
3. Enter “ZGPNativeCurrency” for the attribute name.
4. Select **ENTITIES** for the source dimension.

Example, to set the time period selection to be the current period:

1. Select the **TIMEPERIODS** dimension.
2. Set the attribute class to **SYSTEM**.
3. Enter “SGPCurrentPeriod” for the attribute name.

Specifying post-selection commands

Use post-selection to perform any commands after symbol selection, but before any data areas are created. The typical use of post-selection commands is to perform advanced symbol selection tasks. For example, you might use post-selection to create a list of all standard intercompany accounts. In addition, you may want to use a form to further refine selection, or instead of the template provided symbol selection.

Caution: Post-selection commands are executed as part of symbol selection within a view. Any Create VARIABLE commands will cause an error during symbol reselection. If “Allow user to change selections in view” is allowed, either create variables only in pre-selection, or wrap variable creation in VariableExists checks.

Example: Get a list of all standard intercompany accounts

```
If Not(VariableExists("ACCOUNTS"))
    Create VARIABLE ACCOUNTS AS RANGE
END If
Set VARIABLE ACCOUNTS = CreateList(SYMBOLS, DATABASE, ACCOUNTS, "TB###")
Set VARIABLE ACCOUNTS = FilterList(ACCOUNTS, "
[[SYMBOL,ZElimICSchTransactions,THIS]] == 'ICStandard'")
```

Example: Specifying multiple attribute time periods (current forecast and budget period)

```
If Not(VariableExists("TIMEPERIODS"))
    Create VARIABLE TIMEPERIODS AS RANGE
END If
Set VARIABLE TIMEPERIODS = "[[SYSTEM,SGPForecastPeriod,DBDEFAULT]]" Set
VARIABLE TIMEPERIODS = ListAppend(TIMEPERIODS, "
[[SYSTEM,SGPBudgetPeriod,DBDEFAULT]]")
```

Example: Specify temporary symbols mixed with real symbols

```
If Not(VariableExists("TIMEPERIODS"))
    Create VARIABLE TIMEPERIODS AS RANGE
END If
Set VARIABLE TIMEPERIODS = "[[SYSTEM,SGPForecastPeriod,DBDEFAULT]]"
Set VARIABLE TIMEPERIODS = ListAppend(TIMEPERIODS, DIFF")
Set VARIABLE TIMEPERIODS = ListAppend(TIMEPERIODS, "
[[SYSTEM,SGPBudgetPeriod,DBDEFAULT]]")
```

To capture additional user input via a form, it is required that the button clicked be copied to the FORM_ ButtonClicked variable to ensure expected app execution. Specifically, app execution continues if FORM_ButtonClicked has a value of “OK”.

Example: Capture additional user input via a form

```
Show FORM USING Custom.lvfrm
Set VARIABLE FORM_ButtonClicked = $LVS_BUTTONCLICKED$
```

Allowing currency re-selection in a data view input app

There may be cases where the initial selection in the app is based on the functional currency of the selected entity, but you want the user to be able to change the currency.

To accomplish this, follow these steps:

Note: Substitute CURRENCY for the name of your currency dimension, or use an attribute token or $CORE.DimensionInfo.KeyDimensionList[4]$ for CURRENCY for flexibility.

1. In the pre-selection add the following to support currency selection:

```
//Configure currency selections for re-select in view
Create GLOBALVARIABLE CURRENCYDefault AS STRING
Create GLOBALVARIABLE CURRENCYOptions AS STRING
```

```
Create GLOBALVARIABLE CURRENCYSymbols AS RANGE

Create GLOBALVARIABLE oldSelectDimensionList[] AS STRING //used to
restore symbol selection to original options


Set VARIABLE CURRENCYOptions = "11000"

Set VARIABLE CURRENCYSymbols = "SOURCEC###"


Create VARIABLE isCurrencySelection AS NUM //used to track in post-
selection if the is changing currency only
```

2. In the Post-Selection add logic to set the native currency on entity selection:

```
If Not(VariableExists("CURRENCY"))

    Create VARIABLE CURRENCY AS STRING

END If


If $isCurrencySelection$ //reset selection options to allow regular
symbol selections

    Set VARIABLE FORM_SelectDimensionList = ListAppend(CORE_EmptyList,
oldSelectDimensionList)

    Set VARIABLE isCurrencySelection = 0

Else

    Set VARIABLE CURRENCY = "[[SYMBOL,ZGPNativeCurrency,$ENTITIES$]]"
//apply currency of selected entity

END If
```

3. Add a View Action to allow the user to change currency (this will be a new button on the toolbar)

```
Set VARIABLE oldSelectDimensionList = ListAppend(CORE_EmptyList, FORM_
SelectDimensionList)

Set VARIABLE isCurrencySelection = 1


Set VARIABLE FORM_SelectDimensionList = ListAppend(CORE_EmptyList,
"CURRENCY")


Run PROCEDURE "VIEW\OnReselect.lvpro"
```

Specifying the data area definition

Use Data Area Definitions to specify the data areas to be used in the data input view. For each data area definition to be created, right-click on Data Area Definitions and select New Document.

1. For the Name, enter the name of your Data Area

> **Note:** The name must 29 or fewer characters and only contain alphanumeric characters and underscore "_". The actual name of the data area in application framework will be "da" + Name, so if you need to refer to the Main data area in application framework, use daMain as the name.

2. In the Options table, complete the following fields

Field	Notes
Download this data area	Select this option if you would like to download and display data in this data area. **Note:** ■ If your data area is only being used to lock data, the "Download this data area" option is not required. ■ If you have more than one data area set to download, the last on in the list will be displayed in a single layout data input app.
Submit from this data area	Select this option if you would like to submit from this data area to the database once the submit button is clicked within the data view. **Note:** ■ You cannot set a data area to Submit without enabling the "Download this data area" option as well. ■ At least one data area definition in your data view input app must be set to Submit. ■ Only data that is covered by a lock will be submitted.
Submit comment	Enter the submit comment that will be associated with this data area submission. **Note:** This is only required if "Submit from this data area" is enabled.
Lock this data area	Select this option if you would like to lock this data area. **Note:** The user will be able to enter values for any cells that are downloaded and part of the lock area and to which the user has write access.

Field	Notes
Lock comment	Enter the Lock comment that will be associated with this data area lock. **Note:** This is only required if "Lock this data area" is enabled.

3. In the Data Area table, complete the following fields for each dimension:

Field	Notes
Symbols	Type a symbol name. Alternatively, you can use the symbol selector. To add more than one symbol for a dimension, select a row containing the dimension for which you want to add another symbol and click Symbol. To move a symbol up or down within a dimension, click Move Up or Move Down. **Note:** To use the symbol selected by the user or defined by an attribute, enter the name of the dimension as a variable, for example $ENTITIES$
Spec	Indicate the symbol specification using one of the following values: ■ All: Includes all symbols within the selected symbol's hierarchy. ■ Leaf: Includes only leaf symbols within the selected symbol's hierarchy. ■ Parent: Includes only parent symbols within the selected symbol's hierarchy. ■ Root and Parent: Includes only root and parent symbols within the selected symbol's hierarchy. **Note:** To use the symbol selected by the user or defined by an attribute, enter the name of the dimension as a variable, for example $ENTITIES$
Level	Specify the number of levels down from the symbol to be included in the data area.

4. Use Additional Configuration to specify any other Data Area Definition features you wish to use. Additional Configuration is a code editor that allows you to enter data area definition functions to be included in the data area.

   For more information on the code editor, see Using the code editor.

   Typical use of Additional Configuration in a data area definition would be to

   a. Add an attribute filter using the AttributeFilter function
   b. Add a schedule to the using the Schedule function
   c. Add a temporary symbol using the TempSym function
   d. Override the query type specified in Settings using the AdjustedDetail function.

Specifying the data area views

Use data area views to specify how the data areas will be displayed to the user. For each data area view to be created, right-click on Data Area Views and select New Document.

Note: If using a Single Layout type for your data view, only one Data Area View can be defined.

Name

Specify the name of the data area view

Setting view orientation

Use orientation to determine how each dimension is presented to the user. At least one dimension must be in the rows and columns orientation. Multiple dimensions in rows or columns are nested outer to inner, based on their left to right position.

Use the following table to determine the way dimensions display in the view.

Orientation	View Appearance
Columns or rows	The left-most dimension listed in the orientation section is the outermost dimension in the view.
Slice	Each dimension in the slice dimension is presented to the user as a symbol selector. Dimensions are presented in the view in the order that they appear in the Slice orientation.
Fixed	The view opens displaying fixed symbols in dimension order on the Fixed Symbols tab. Dimensions appear on the Fixed Symbols tab in their natural order regardless of the order listed in the Fixed orientation.

To modify dimension orientation:

1. In the Orientation section for the dimension you want to reorient, select the dimension.
2. Drag and drop the selected dimension into the desired position.

Specifying symbols

Use symbol selection to specify the symbols that are displayed in the view. In the Symbol Selection table, complete the following fields for each dimension:

Note: You can copy the symbol selections from a particular Data Area Definition into the Symbol Selection table by selecting the Data Area Definition in the from drop list and clicking the Copy Default Selections button. For dimensions set as fixed in the orientation, the first symbol specified in the data area will be copied into the selections. The Spec and Level fields are not applicable if a symbol contains a variable.

Field	Notes
Symbol	Type a symbol name. Alternatively, you can use the symbol selector. If you specified a data area definition for the view, symbol selections are limited by the it. To add more than one symbol for a dimension, select a row containing the dimension for which you want to add another symbol and click Symbol. To move a symbol up or down within a dimension, click Move Up or Move Down. **Note:** To use the symbol selected by the user or defined by an attribute, enter the name of the dimension as a variable, for example $ENTITIES$
Spec	Indicate the symbol specification using one of the following values: ■ All — Includes all symbols within the selected symbol's hierarchy. ■ Leaf — Includes only leaf symbols within the selected symbol's hierarchy. ■ Parent — Includes only parent symbols within the selected symbol's hierarchy. ■ Root and Parent — Includes only root and parent symbols within the selected symbol's hierarchy.
Level	Specify the number of levels down from the symbol to be included in the data area.
Levels to expand	For dimensions in the rows and column orientation only, you can specify the number of levels to expand the hierarchy of each symbol included.

Specifying dimension display settings

Use dimension display settings to specify how symbols are displayed in the view.

In the Dimension Display Settings table, complete the following fields for each dimension:

Note: Show Name and Show Description only affect dimensions in the Rows or Columns orientation.

Field	Notes
Show name	Check this to display the name of the symbol in the view. If both Show Name and Show Description are selected the symbol name and description will be displayed separated by a hyphen for symbols in the dimension.
Show description	Check this to display the description of the symbol in the view. If both Show Name and Show Description are selected the symbol name and description will be displayed separated by a hyphen for symbols in the dimension.

Field	Notes
v26.2 Alternate Description	Select this to display an alternate description for the symbol in the view based on a symbol's attribute value. Choose from the Symbol Attributes drop-down (default: None). When selected, the symbol attribute value of each symbol is displayed (if set). If Show Name and /or Show Description are selected, the attribute value is displayed after them, separated by hyphens.
Fit to tile	Check this to make the view auto-size to fit symbol titles. For Rows this affects the width of the title. For Columns this can only be selected for the inner dimension and affects the width of the column.
Column width	If Fit to Title is not selected, enter the width in pixels.
Wrap	If Fit to Title is not selected, check this to wrap titles onto new lines if the text does not fit in the specified width.

Specifying number formatting

Use Number Formatting to specify how numbers are displayed in the view. In the Number Formatting table complete the following fields:

Field	Notes
Default decimals	Enter the number of decimals to display for each numeric value. Generally, this should be from 0 to 9. This can be overridden for a specific symbol using the SymbolDecimals function in the Additional Configuration section.
Use parentheses for negative numbers	Select Yes to display negative numbers with parentheses. Select No to display negative numbers with a negative sign.
Use thousands separator for numbers	Select Yes to display a locale-specific thousands separator based on the operating system setting. Select No to not display a thousand separator.

Specifying the left title

Use Left Title to optionally specify the left title that appears in the view.

Field	Notes
Left title	Enter text to display in the left title area of the view. This is optional and can be left blank.

Field	Notes
Include entity in title	Check this to include the entity in the left title area of the view. Entities can only be included in the title under one of the following conditions. ■ Entities must be a dimension specified as a user selection and that selection must not allow multiple selections, or ■ Entities must be a dimension specified as an attribute-based selection and the source dimension (SYMBOL class) must not allow multiple selections, or ■ A variable must be created as the name of the dimension with a type STRING. This can be done in the Pre or Post Selection sections of the DVI.
Include currency in title	Check this to include the currency in the left title area of the view. Currencies can only be included in the title under one of the following conditions. ■ Currencies must be a dimension specified as a user selection and that selection must not allow multiple selections, or ■ Currencies must be a dimension specified as an attribute-based selection and the source dimension (SYMBOL class) must not allow multiple selections, or ■ A variable must be created as the name of the dimension with a type STRING. This can be done in the Pre or Post Selection sections of the DVI.
Include time period in title	Check this to include the time period in the left title area of the view. Time Periods can only be included in the title under one of the following conditions. ■ Time Periods must be a dimension specified as a user selection and that selection must not allow multiple selections, or ■ Time Periods must be a dimension specified as an attribute-based selection and the source dimension (SYMBOL class) must not allow multiple selections, or ■ A variable must be created as the name of the dimension with a type STRING. This can be done in the Pre or Post Selection sections of the DVI.

Specifying options

Use Options to specify additional view settings. In the Options table complete the following fields:

Field	Notes
Allow user to change selections in view	Select Yes to allow the user to change selections once the view is displayed. This allows the user to update the user selections without closing the app and re-opening it. **Note:** Select No if no user selections are defined in the app.

Field	Notes
Allow dynamic rollup of input	Select Yes to have parent totals automatically calculated as the user enters numeric values.
Use on demand calculations	Select Yes to allow the user to run on demand calculations. An additional icon Calculate will be added to the toolbar to allow the user to perform the calculation(s) set to run on demand.
Dynamic Calculations	Select calculations to be performed as the user manually changes values in the view. The selections available are determined by the contents of the Calculations folder. **Note:** Calculations are executed in the order listed.
Use on demand validations	If validations are defined for the input app, select Yes to display a toolbar icon to allow the user to run validations on demand. Validations are still executed on save regardless of this setting.

Selecting tools

Use Tools to enable the use of input tools within the view. See Input tools for information on the input tools. In the Tools table complete the following fields:

Field	Notes
Allow use of adjust values tool	Select Yes to allow the user to right-click on a leaf cell to invoke the adjust values tool.
Allow use of copy across tool	Select Yes to allow the user to right-click on a leaf cell to copy the value across to all other leaf cells under the root in the view.
Allow use of pattern spread tool	Select Yes to allow the user to right-click on a parent cell and use the pattern spread tool to populate all leaf cells under the parent using a pattern.
Allow use of quick input tool	Select Yes to allow the user to right-click on a parent cell and use the quick input tool to populate all leaf cells under the parent.
Allow use of quick spread tool	Select Yes to allow the user to right-click on a parent cell and use the quick spread tool to populate all leaf cells under the parent using a trend.
Allow use of import	Select Yes to allow the user to import data from a file. The file to be imported must specify the symbol in each dimension and the value. The user will be able to clear all data in the view and select the field delimiter used in the file.
Allow use show underlying adjustments	Select Yes to allow the user to access underlying adjustments via the cell context menu. Any adjustments impacted the selected cell will be displayed in the Show Adjustments table

Field	Notes
Allow use show audit trail	Select Yes to allow the user to access audit trail via the cell context menu. Any data changes impacting the selected cell will be displayed in the Show Audit Trail table.
Allow use show underlying comments	Select Yes to allow the user to access underlying comments via the cell context menu and toolbar. Any comments related to the selected cell will be displayed in the Show Comments table. If executed from the toolbar, all comments related to the entire data area will be displayed in the Show Comments table.

Customizing the data view

Use Additional Configuration to specify any other view features you wish to use in the data view. Additional Configuration is a code editor that allows you to enter view functions to be included in the configuration of the view.

For more information on the code editor, see Using the code editor.

Typical use of Additional Configuration in this app would be to

- Add protected areas to the view using the Protect function
- Add conditional style to the view using the ConditionalStyle function
- Restrict input to numeric values or text values using the NumericInputOnly and/ or TextInputOnly functions
- Hiding specific symbols in the view using the Hide function
- Overriding the decimals displayed for certain symbols using the SymbolDecimals function
- Adding separators between symbols using the InsertSeparator function.

Specifying view actions

Use Actions to specify custom toolbar and context menu actions with a view. For each action to be created, right-click on Actions and select New Document.

For each action complete the following fields:

Field	Notes
Name	Enter the name of the action.
Action location	Select whether the action will appear in the toolbar or on the cell context menu.
Views to add action to	Select the data area views to assign the action to.

Field	Notes
Action text	Enter text to display for the action. For a toolbar location, the text will appear to the right of the icon, if specified. For a context menu this will be the menu text. **Note:** This is required for a context action, but is only required for a toolbar action, if an icon is not specified.
Action icon	Enter the path to the icon to display for the action **Note:** This is optional for a context action but is required for a toolbar if action text is not specified.
Action tooltip text	Enter additional information that will appear when the user hovers the mouse on the action. This is optional.
Cell restriction expression	Enter an expression to limit the cells the context menu appears on. This is optional and only applied when the action location is Cell. You can use Not, And, and Or in conjunction with PARENT, LEAF, STATIC, WRITABLE, and READONLY. For more information on the expression, see Action in the *Longview Developer's Guide*.
Action	Enter the commands to execute for this action via the code editor. For more information on the code editor, see Using the code editor.

Specifying tabs

Use Tabs to add tabs to the data grid. Each tab can display a different data area view. For each Tab to be created, right-click on Tabs and select Add Tab.

Note: Tabs are only required if using a Tabbed Layout Type. For more information, see Specifying overall settings.

Field	Notes
Data area	Select the data area definition to be used with this tab.
Data view name	Select the data area view to be used with this tab.
Tab name	Use this to provide a label to display on the tab
Action to run when switching to this tab	Select the command to execute when this tab is switched to by the user. The selections available are determined by the contents of the Tab Commands folder. This parameter is optional. **Note:** If this is the first tab this command will be executed when the view first appears.
Action to run when switching from this tab	Select the command to execute when this tab is switched from by the user. The selections available are determined by the contents of the Tab Commands folder. This parameter is optional.
Background color	Use this to select the background color for the tab. The default color is transparent.

Specifying post-refresh commands

Use Post-Refresh to perform any commands after the data areas are downloaded but before any refresh calculations are performed. Use Post-Refresh to perform any processing that requires data to be downloaded into the data areas.

Note: Post-refresh commands are executed prior to any calculations with the “Run calculation when data is refreshed” option.

Specifying a tab action

Use Tab Actions to define commands and functions to be executed when switching between tabs in a tabbed view.

For more information on the code editor, see Using the code editor.

Specifying calculations

Use calculations to create models that are executed as part of the input app. Each model can be executed at various points in the app and is executed in the order listed in the Calculations folder. Use the calculation code block to specify the calculations using the code editor.

For more information on using the code editor, see Using the code editor.

For each model specify the run properties:

Field	Notes
Data areas to run calculation on	Select the data area definitions to which this calculation applies.
Run calculation when data is refreshed	Check this to cause the calculation to occur when data is downloaded initially and whenever the user clicks Refresh.
Prompt for submit on close	If run calculation when data is refresh is enabled, check this option to indicate that the values calculated will be treated as data changes and submitted to the database. The user will be prompted to submit changes when the view is closed even if no other changes were made.
Run calculation after user imports data	Check this to cause the calculation to occur after the user imports data via the import tool. **Note:** Allow use of import tool must be enabled.
Run calculations after user uses an input tool	Check this to cause the calculation to occur after the user enters data using an input tool. **Note:** At least one input tool must be enabled.

Field	Notes
Run calculation before data is submitted	Check this to cause the calculation to occur when the user clicks Submit, but before the data is uploaded. **Note:** This calculation will only be executed if other data has been changed in prior to submission.
Run calculation after data is submitted	Check this to cause the calculation to occur when the user clicks Submit, but after the data is uploaded. This is mostly useful when the calculation before submitting changes the data in ways that are confusing to a user remaining in the app, for example clearing data that should not be submitted.
Run calculation on demand	Check this to cause the calculation to occur when the user clicks Calculate.

Specifying data validations

Use Validations to specify data validations to perform when the user clicks the submit or validate icon. Validation failures can be specified as errors or warnings. Errors will prevent the user from submitting data, while warnings will inform the user of validation failures, but still allow the user to proceed with submitting data.

Note: For the Data View Input app there is a single validation procedure for all data views. Be sure to reference the correct data area name in any validation functions.

Field	Notes
Validation dimensions	Use this to specify dimensions to include in validation results. This is normally used in cases where slice dimensions are used in the view to give the user better feedback as to where the data validation failure occurred.

Repeat for each validation:

1. Create a variable to store the validation result.

   Note: You can use the same result variable for each validation.

2. Write an expression to determine the validation result. Store the result in the variable created in step 1.
3. Write a condition to check the validation result against the expected result.
4. If the validation result does not pass the check, Set the VALD_Result variable and run procedure VALD\AddResult.lvpro to add the validation failure result to the list of validation failures.

   Note: You can type in Template_Validation into the validation area in your Data View Input app to get the sample syntax code below.

Example:

```
Create VARIABLE val as NUM
Set VARIABLE val = Value("daMain", "BS", "$TIMEPERIODS$", "$ENTITIES$",
"GL", "SCENARIOS_Working", "$CURRENCIES$")
If $val$ != 0
    Set VARIABLE VALD_Result = "ERROR|Balance sheet is out of balance|$val$"

    Run PROCEDURE "VALD\AddResult.lvpro"
END If
```

Setting variable VALD_Result

The VALD_Result variable is a string list variable, set as follows (separate each item with a pipe character “|”):

1. ERROR or WARNING to specify the validation failure level
2. The validation failure message to display to the user
3. The validation failure amount
4. If validation dimensions are specified, the symbol for each dimension

Specifying forms

Use Forms to capture user input:

1. In an action within an input template
2. During post-selection in an app

For each form complete the following fields:

Field	Notes
Name	Enter the name of the form.
Title	Enter the title for the form. This is optional. If left blank the title of the form will be the same as the description of the app.
Width (in pixels)	Enter the width of the form. If not specified, the default width of 450 is used.
Height (in pixels)	Enter the height of the form. If not specified, the height is determined by the number of controls in the form.
Format for the date value returned	Enter the format a date is stored in an application framework variable when returned from a form. The format in the form is determined by the user’s regional settings.
Form controls	Add the controls to be displayed in the form via the code editor. For more information on the code editor, see Using the code editor.

Specifying instructions

Use Instructions to specify instructions to be displayed in the instructions tab in the view. Instructions are specified using HTML. A general layout for instructions is provided. For each instruction page to be

created, right-click on Instructions and select New Document.

1. Add list items <li> to the Data Entry list to specify data entry instructions.
2. Add list items <li> to the Note list to specify any notes related to the input.
3. Modify the Comments section as required to remove any features not used in the app.

Creating Data Import Apps

You can use Longview Designer to create data import apps. Data import apps can be used to import data into your system.

Note: If you intend to run your data import app in batch mode, include only commands supported in batch mode; do not include any Show commands.

Working with variables

You can use Longview Designer to create, modify, reorder, or delete variables for your app. Variables are optional. You might include variables in your app if you want to, for example, use an attribute such as the current time period, or if you want to prompt a user to select a symbol or a file.

Once you have created variables, you can use them elsewhere in your app by enclosing them in dollar signs ($ $).

For more information, see “Variables” in the *Longview Developer’s Guide*.

You can also use external variables not defined on the Variables page of your app by enclosing them in dollar signs ($ $).

Variables tasks include:

- Creating variables
- Modifying variables
- Reordering variables
- Deleting variables

Creating variables

You can use Longview Designer to create variables for your app. Each variable you specify is created and set to the specified value when your app runs.

To create a variable:

1. Open the Variables section.
2. Do one of the following:
   - To create a new variable at the bottom of the list, click Variable. The new variable appears at the bottom of the variables list.

- To create a new variable below a specific variable, select a row and click Variable. The new variable appears below the selected row.

3. Complete the following fields:

Field	Notes
Name	Type the name of the variable. Do not prefix the variable name with “LV_” or “LVS_”.
Type	Specify the type of data to be stored in the variable using one of the following options: ■ String — Designates a string value. ■ Range — Designates a list of symbols. ■ Number — Designates a numeric value. ■ List — Designates a list of string values. ■ NumberList — Designates a list of numeric values. The default is String.
Expression	Specify the value that the variable is set to. Based on the Type field this can be a number, a string, or a list of symbols, respectively.

Note: You can include attribute values in an expression. To include an attribute value in an expression, you must first set a variable to the value of the attribute using an attribute token, and then use the variable. For example, you could create a string variable named “currentPeriod” and use the expression “[[SYSTEM,SGPCurrentPeriod,DBDefault]]”. For more information on setting variables, see “Set Variable” in the *Longview Developer’s Guide*. For more information on using attributes, see “Attribute tokens” in the *Longview Developer’s Guide*.

Modifying variables

You can use Longview Designer to modify existing variables for your app.

To modify a variable:

1. Open the Variables section.
2. Select the variable you want to modify and make the necessary changes.

Reordering variables

You can use Longview Designer to change the order of variables in your app. The order of variables is important because you cannot use a variable in an expression until you first define the variable. Variables are created in the same order in which they are defined.

To reorder a variable:

1. Open the Variables section.
2. Select the variable you want to reorder, and then click either Move Up or Move Down. The variable moves one row in the indicated direction.
3. If you want to continue to move the selected variable, repeat step 2 until the variable is in the desired location.

Deleting variables

You can use Longview Designer to delete variables from your app.

To delete a variable:

1. Open the Variables section.
2. In the variables list, select the variable you want to delete, and click Delete. The variable is deleted.

Specifying import file details

You can use Longview Designer to specify import file details. The import file contains the source data you want to import.

To specify import file details:

1. In the Data Import Apps category, expand Existing Apps. A list of existing data import apps displays.
2. Open the data import app for which you want to specify import file details. The data import app opens in the workspace with the Properties page in view.
3. Click the Import File tab.

4. Complete the following fields:

Field	Notes
File	Specify the path and name of the import file.
Number of Header Records to Filter	Specify the number of header rows to filter on import. If you leave the value as the default value of 0, the system assumes your import file starts with data records with no header rows.
Number of Footer Records to Filter	Specify the number of footer rows to filter on import. If you leave the value as the default value of 0, the system assumes your import file ends with data records with no footer rows.
Field Delimiter	Specify whether the character used to separate fields in the import file is the brace ({), comma (,), semicolon (;), or pipe (|). Alternatively, you can type a field delimiter of your choice. The default is the brace ({).
Number of Fields	Specify the number of fields per line in the import file using an integer from 1 through 999.
Number of Value Fields	Specify whether the import file contains a single value field per line or multiple value fields per line and the corresponding details using one of the following values: ■ Single Value at Field Position — Indicates there is a single value field in each line of the import file. For example, each line of your import file may contain a data value in the third field position. If you select this option, you must specify the field position at which the data value is found using a number from 1 up to and including the number you specified for Number of Fields. For example, type 3. ■ Multiple Value Fields for Dimension — Indicates there are multiple value fields in each line of the import file, with each value corresponding to a different symbol in the specified dimension. For example, in the TIMEPERIODS dimension you may have eight data values in each line of the import file corresponding to eight symbols in the TIMEPERIODS dimension. If you select this option, you must specify the dimension for which there are multiple value fields. For example, select TIMEPERIODS from the list.

5. In the Dimension Definition section, complete the following fields for each dimension:

Field	Notes
Schedule	This field displays only if your system is configured to use schedules and you have access to schedules. The default of None indicates the data import app is on base data only. Specify the schedule to use for the data import app, and the schedule dimensions appear at the bottom of the list of Dimensions.

Field	Notes
Type	Specify the type of relationship between dimension symbols and field values in the import file using one of the following values: ■ Consecutive — The values encountered in the import file are placed consecutively in the dimension starting at the indicated symbol, moving in priority order through the indicated symbol's siblings. **Note:** Consecutive is designed to work with a symbol existing in only one hierarchy. If the symbol belongs to more than one hierarchy, the behavior used to determine the siblings is undefined and may be affected by hierarchy reorganizations and imports/exports. You can specify only one dimension as Consecutive. ■ Map — The values encountered in the import file are mapped to dimension symbols as specified in the indicated map. ■ Match — The values encountered in the import file are matched to the dimension symbols exactly. ■ Unique — All values encountered in the import file are placed in the specified dimension symbol. If you select this option, you can use the variables you defined on the Variables page for the specified symbol. **Note:** If you selected Multiple Value Fields for Dimension in step 4, the dimension you indicated has Values for the Type field and cannot be changed.
Field Position	This field is available only when Type is Map or Match. Specify the field position to assign the dimension to using a positive integer. For example, for the Accounts dimension, type 3 if the account will be the third column of the data target.
Symbol	This field is available only when Type is Unique. Type the symbol name for the specified dimension to which the data is imported. Alternatively, you can use the symbol selector. **Note:** You can also use a variable to define the symbol.

Field	Notes
Map Location	This field is available only when Type is Map. Indicate the location of the map using one of the following values: ■ Internal — Indicates the map exists inside the database, created using the Mappings editor. For more information, see "Maintaining mappings". ■ External — Indicates the map exists as a file, such as a text file on your local machine or network location. ■ Document — Indicates the map exists as a document within the Symbol Maps folder of the app.
Map	This field is available only when Type is Map. Do one of the following: ■ If you selected Internal as the Map Location, select a map from the list of available maps in the system. ■ If you selected External as the Map Location, type the path of the external map or click the ellipsis button (...) to select the external map. **Note:** You can use a variable for the Map field value.

6. Do one of the following:
   - If you selected Single Value at Field Position in step 4, proceed to step 9.
   - If you selected Multiple Value Fields for Dimension in step 4, click Value Field in the Value Fields section to add a new Value Field. A new row appears in the table.
7. Complete the following fields:

Field	Notes
Field Position	Specify the field position of the value field using a positive integer, where the field position is the corresponding column in the data source containing the data. For example, if the value is exported to the third column in the target file, type 3.
Symbol	Type the name of a symbol from the dimension indicated in Multiple Value Fields for Dimension for the value in the corresponding field position. Alternatively, you can use the symbol selector. **Note:** You can use a variable for the Symbol field.

Note: You can delete a value field by selecting a value field row and clicking Delete.

8. Repeat step 6 and step 7 for each value field in the import file.

9. Optionally, to apply a filter to the records in the import file where records meeting the filter criteria are the only records imported, click Filter in the Filters section to add a new Filter. A new row appears in the table.

   Note: You can delete a filter by selecting a filter row and clicking Delete.

10. Type an expression for the filter using the following guidelines:

    - The expression can be joined together by two or more expressions using AND or OR and grouped by brackets, as appropriate.
    - Each expression uses field names, operators, and values as follows: FieldName EQ|NE|LE|LT|GE|GT Value. If Value is a string, enclose the string in double quotation marks.
    - You can optionally include wildcard characters (? and *) and symbols for Value.
    - You can optionally use symbols for operators, such as ==, !=, <=, <, >=, and >.
    - You must use Field# for FieldName to refer to a specific field on which you want to filter. For example, if Products is the fourth column in your import file and you want to filter Products to import only the Products_Default symbol, use Field4 == "Products_Default" as the expression.

Note: For more information on creating expressions, see “Using conditional operators” and “Search” in the *Longview Developer’s Guide*.

Specifying data areas

You can use Longview Designer to define the data area specification and related information for your data import app. The data area specification defines the data area to lock and to which the data is imported. If the data area includes a schedule, all symbols in the schedule dimensions are included in the data area specification.

To specify a data area:

1. In the Data Import Apps category, expand Existing Apps. A list of existing data import apps displays.
2. Open the data import app for which you want to specify a data area. The data import app opens in the workspace with the Properties page in view.
3. Click the Data Area tab.

4. Complete the following fields:

Field	Notes
Lock description	Type a description for the data area lock. The lock description can have a maximum of 100 characters, including spaces. The default is Data Import - name of data import app. **Note:** You can use the variables you defined on the Variables page for the Lock Description field.
Submission description	Type a description for the submission. The submission description can have a maximum of 100 characters, including spaces. The default is Data Import - name of data import app. **Note:** You can use the variables you defined on the Variables page for the Submission Description field.
Schedule	This field displays only if your system is configured to use schedules and you have access to schedules. This field displays the schedule selected for the data import app in Specifying import file details.

5. In the table, complete the following fields for each dimension for the data area specification of the data import:

Note: The Spec and Level fields are not applicable if a symbol contains a variable.

Field	Notes
Symbols	Type a symbol name. Alternatively, you can use the symbol selector. To add more than one symbol for a dimension, select a row containing the dimension for which you want to add another symbol and click Symbol. To move a symbol up or down within a dimension, click Move Up or Move Down. **Note:** To use the symbol selected by the user or defined by an attribute, enter the name of the dimension as a variable, for example $ENTITIES$

Field	Notes
Spec	Indicate the symbol specification using one of the following values: ■ All: Includes all symbols within the selected symbol's hierarchy. ■ Leaf: Includes only leaf symbols within the selected symbol's hierarchy. ■ Parent: Includes only parent symbols within the selected symbol's hierarchy. ■ Root and Parent: Includes only root and parent symbols within the selected symbol's hierarchy. **Note:** To use the symbol selected by the user or defined by an attribute, enter the name of the dimension as a variable, for example $ENTITIES$
Level	Specify the number of levels down from the symbol to be included in the data area.

Specifying custom code

You can use Longview Designer to specify custom code for your data import app. Longview provides you with the following insertion points in the import process where you can specify custom code:

Custom Code area	Use this area to...
Pre-Import	specify the code to run before the import file is processed.
Post-Import	specify the code to run after the import file is processed.
Post-Upload	specify the code to run after the data is uploaded.

You can set the LV_Success variable to 0 at any point within your custom code to indicate an error and stop the import process. You may choose to, for example, specify custom code in order to validate the data before the upload occurs.

To specify custom code:

1. In the Data Import Apps category, expand Existing Apps. A list of existing data import apps displays.
2. Open the data import app for which you want to specify custom logic. The data import app opens in the workspace with the Properties page in view.
3. Click the Custom Code tab.
4. In the relevant custom code area, specify your code as needed.

Specifying options

You can use Longview Designer to specify options for your data import app.

To specify options:

1. In the Data Import Apps category, expand Existing Apps. A list of existing data import apps displays.
2. Open the data import app for which you want to specify options. The data import app opens in the workspace with the Properties page in view.
3. Click the Options tab.
4. Complete the following fields:

Field	Notes
Data Import Method	Specify the way the data is imported using one of the following values: ■ Full — The data area is not downloaded before the system imports the data. The system imports zeros to any cells in the data area not included in the import file. ■ Incremental — The data area is downloaded before the system imports the data. The system does not import values to any cells in the data area not included in the import file. The default is Full.
Decimal Character	Specify the decimal character for the data import app using one of the following values: ■ " . " — The period is the decimal character. ■ " , " — The comma is the decimal character. The default is the period " . ".
Apply Sign Reversal Logic	Select this option to reverse the sign for numeric values for accounts with Credit as the Balance Type. For more information on the balance type in the ACCOUNTS dimension, see "Working with dimensions" in the *Longview Application Administrator Guide*. The default is cleared.

Field	Notes
Allow Duplicate Mappings	This option applies only when Type is Map. Select this option to indicate that duplicate mappings are permitted. If multiple mappings are found, the system does the following for each type of mapping: ■ Exact — All duplicate Exact mappings are used so that the same value is imported to multiple data cells. If there are duplicate Exact and Wildcard or Range mappings, the Exact mapping is used and the Wildcard or Range mapping is ignored. ■ Wildcard — If no Exact mappings are found, the value is imported to the first Wildcard or Range mapping listed. ■ Range — If no Exact mappings are found, the value is imported to the first Wildcard or Range mapping listed. If you do not select this option, duplicate mappings are not permitted. If multiple mappings are found, the system reports an error for each mapping. The default is cleared. For more information on duplicate mappings, see Understanding duplicate mappings.
Duplicate Records	Specify the way duplicate records are handled for the data import app using one of the following values: ■ Disallow — Specifies that duplicate records should not be allowed. The first entry is submitted. If a duplicate record is encountered, the system reports an error. ■ Add — Specifies that duplicate records should be allowed and the value of all such records summed. If a duplicate record is encountered that contains text (versus numeric) data, it is treated as an invalid/error record. ■ Use First — Specifies that the first entry of duplicate records should be used. If a duplicate record is encountered, the system does not report an error. ■ Use Last — Specifies that the last entry of duplicate records should be used. If a duplicate record is encountered, the system does not report an error. The default is Disallow.
Maximum Errors Allowed	Specify the maximum number of error records to permit before stopping the import process. The default is 0.

Field	Notes
Prompt Before Importing	Select this option to display a prompt dialog to the user before the data is imported, and optionally customize the message displayed to the user. The prompt dialog contains the buttons Import and Cancel. The default message displayed to the user is “Are you sure you want to import data?”. **Note:** Variables created on the Variables page of this data import app are not supported for the message displayed to the user. If you do not select this option, the user does not see a message before data is imported. The default is cleared.
Error File	Specify the path and name of the error file. This creates a text file in the location specified. All rejected records are captured in this file. **Note:** You can use the variables you defined on the Variables page for the Error File field.
Log File	Specify the path and name of the log file. This creates a text file in the location specified. This log file captures all information related to the import such as the name and value of all variables created during execution. Error codes returned to the log file for ‘process complete’ or ‘terminated’ include: ■ invalid coordinates encountered (i.e. parent symbols) ■ duplicate records detected ■ line number and error if a line of data fails ■ aborted with errors (for example, if the MAXERROR parameter has been reached) ■ completed with errors ■ completed without errors (successful) ■ process not complete **Note:** You can use the variables you defined on the Variables page for the Log File field.

Creating Single Value Data Area Export Apps

You can use Longview Designer to create single value data area export apps.

Working with variables

You can use Longview Designer to create, modify, reorder, or delete variables for your app. Variables are optional. You might include variables in your app if you want to, for example, use an attribute such as the current time period, or if you want to prompt a user to select a symbol or a file.

Once you have created variables, you can use them elsewhere in your app by enclosing them in dollar signs ($ $).

For more information, see “Variables” in the *Longview Developer’s Guide*.

You can also use external variables not defined on the Variables page of your app by enclosing them in dollar signs ($ $).

Variables tasks include:

- Creating variables
- Modifying variables
- Reordering variables
- Deleting variables

Creating variables

You can use Longview Designer to create variables for your app. Each variable you specify is created and set to the specified value when your app runs.

To create a variable:

1. Open the Variables section.
2. Do one of the following:
   - To create a new variable at the bottom of the list, click Variable. The new variable appears at the bottom of the variables list.
   - To create a new variable below a specific variable, select a row and click Variable. The new variable appears below the selected row.
3. Complete the following fields:

Field	Notes
Name	Type the name of the variable. Do not prefix the variable name with “LV_” or “LVS_”.

Field	Notes
Type	Specify the type of data to be stored in the variable using one of the following options: ■ String — Designates a string value. ■ Range — Designates a list of symbols. ■ Number — Designates a numeric value. ■ List — Designates a list of string values. ■ NumberList — Designates a list of numeric values. The default is String.
Expression	Specify the value that the variable is set to. Based on the Type field this can be a number, a string, or a list of symbols, respectively.

Note: You can include attribute values in an expression. To include an attribute value in an expression, you must first set a variable to the value of the attribute using an attribute token, and then use the variable. For example, you could create a string variable named "currentPeriod" and use the expression "[[SYSTEM,SGPCurrentPeriod,DBDefault]]". For more information on setting variables, see "Set Variable" in the *Longview Developer's Guide*. For more information on using attributes, see "Attribute tokens" in the *Longview Developer's Guide*.

Modifying variables

You can use Longview Designer to modify existing variables for your app.

To modify a variable:

1. Open the Variables section.
2. Select the variable you want to modify and make the necessary changes.

Reordering variables

You can use Longview Designer to change the order of variables in your app. The order of variables is important because you cannot use a variable in an expression until you first define the variable. Variables are created in the same order in which they are defined.

To reorder a variable:

1. Open the Variables section.
2. Select the variable you want to reorder, and then click either Move Up or Move Down. The variable moves one row in the indicated direction.
3. If you want to continue to move the selected variable, repeat step 2 until the variable is in the desired location.

Deleting variables

You can use Longview Designer to delete variables from your app.

To delete a variable:

1. Open the Variables section.
2. In the variables list, select the variable you want to delete, and click Delete. The variable is deleted.

Specifying pre-selection commands

Use pre-selection to perform any commands before symbol selection. The typical use of pre-selection is to perform advanced logic to determine default selections or set the range of possible symbols. Pre-selection commands are executed after symbol selection settings are applied from the app configuration, so all symbol selection variables have been created at this point.

See Generating a symbol selection form for details on symbol selection variables.

Note: Using pre-select commands to set default or symbols overrides the settings in User Selections.

Variable	Notes
<Dimension>Default	Specify the default selection in the symbol selector, for the Dimension.
<Dimension>Symbols	Specify the list of available selections in the symbol selector for the Dimension.

Example: Setting the list of possible symbols to all YTD periods

```
Create VARIABLE ytdPeriodList AS RANGE
Set VARIABLE ytdPeriodList = "[[SYSTEM,SGPTimePeriodsYTD,DBDEFAULT]]"
For EACH ytdPeriod in ytdPeriodList
    Set VARIABLE TIMEPERIODSSymbols = ListAppend(TIMEPERIODSSymbols,
"$ytdPeriod$#99")
Next
```

Specifying user selections

Use user selections to specify dimensions for which the user will be prompted to choose symbol(s). For each dimension that the user will choose symbols, right-click on the User Selections folder and choose Add Dimension. The dimension settings page will appear.

Variable	Notes
Dimension	Select the dimension the user will be prompted to select symbol(s) for.
Available selections	Select a symbol to limit the possible symbols available to the user. If left blank the user will be able to select from all symbols available in the dimension. The available symbols will always be limited by user access.

Variable	Notes
Default selection	Select a symbol to specify as the default selection. This can either be a database symbol, an attribute symbol, or a floating time period. **Note:** Attribute symbols and floating time periods will only be available if they are defined in your system. For more information, see “Specifying attributes in Longview Application Administrator” in the *Longview Analysis and Reporting Guide*.
Selection required	Check this to force the user to select a symbol to continue.
Allow leaf selection	Check this to allow the user to select a leaf symbol.
Allow parent selection	Check this to allow the user to select a parent symbol.
Allow read-only selection	Check this to allow the user to select a read-only symbol.
Allow multiple selections	Check this to allow the user to select more than one symbol.
Attribute filter	Enter an attribute filter expression to further limit the possible symbol selections. For more information see SymbolSelector in the *Longview Developer’s Guide*.

Specifying attribute-based selections

Use Attribute Based Selections to specify selections for dimensions that are deter- mined by an attribute value. For each dimension to be determined via an attribute, right-click on Attribute Based Selections and select Add Dimension.

Variable	Notes
Dimension	Select the dimension to select symbol(s) using an attribute value.
Attribute class	Select the class of attribute to use.
Attribute name	Select the attribute.
Source dimension (SYMBOL class)	If the attribute class is SYMBOL, select the source dimension. **Note:** The source dimension must be a dimension specified as a user selection or attribute-based selection.

Note: The variable created to hold the attribute-based symbol will have the same name as the dimension. The variable created will be a STRING type.

Example, to set the currency selection to be the functional currency of the selected entity:

1. Select the **CURRENCIES** dimension.
2. Set the attribute class to **SYMBOL**.
3. Enter “ZGPNativeCurrency” for the attribute name.
4. Select **ENTITIES** for the source dimension.

Example, to set the time period selection to be the current period:

1. Select the **TIMEPERIODS** dimension.
2. Set the attribute class to **SYSTEM**.
3. Enter “SGPCurrentPeriod” for the attribute name.

Specifying post-selection commands

Use post-selection to perform any commands after symbol selection, but before any data areas are created. The typical use of post-selection commands is to perform advanced symbol selection tasks. For example, you might use post-selection to create a list of all standard intercompany accounts. In addition, you may want to use a form to further refine selection, or instead of the template provided symbol selection.

Caution: Post-selection commands are executed as part of symbol selection within a view. Any Create VARIABLE commands will cause an error during symbol reselection. If “Allow user to change selections in view” is allowed, either create variables only in pre-selection, or wrap variable creation in VariableExists checks.

Example: Get a list of all standard intercompany accounts

```
If Not(VariableExists("ACCOUNTS"))
    Create VARIABLE ACCOUNTS AS RANGE
END If
Set VARIABLE ACCOUNTS = CreateList(SYMBOLS, DATABASE, ACCOUNTS, "TB###")
Set VARIABLE ACCOUNTS = FilterList(ACCOUNTS, "
[[SYMBOL,ZElimICSchTransactions,THIS]] == 'ICStandard'")
```

Example: Specifying multiple attribute time periods (current forecast and budget period)

```
If Not(VariableExists("TIMEPERIODS"))
    Create VARIABLE TIMEPERIODS AS RANGE
END If
Set VARIABLE TIMEPERIODS = "[[SYSTEM,SGPForecastPeriod,DBDEFAULT]]" Set
VARIABLE TIMEPERIODS = ListAppend(TIMEPERIODS, "
[[SYSTEM,SGPBudgetPeriod,DBDEFAULT]]")
```

Example: Specify temporary symbols mixed with real symbols

```
If Not(VariableExists("TIMEPERIODS"))
    Create VARIABLE TIMEPERIODS AS RANGE
END If
Set VARIABLE TIMEPERIODS = "[[SYSTEM,SGPForecastPeriod,DBDEFAULT]]"
Set VARIABLE TIMEPERIODS = ListAppend(TIMEPERIODS, DIFF")
Set VARIABLE TIMEPERIODS = ListAppend(TIMEPERIODS, "
[[SYSTEM,SGPBudgetPeriod,DBDEFAULT]]")
```

To capture additional user input via a form, it is required that the button clicked be copied to the FORM_ ButtonClicked variable to ensure expected app execution. Specifically, app execution continues if FORM_ButtonClicked has a value of “OK”.

Example: Capture additional user input via a form

```
Show FORM USING Custom.lvfrm
Set VARIABLE FORM_ButtonClicked = $LVS_BUTTONCLICKED$
```

Allowing currency re-selection in a data view input app

There may be cases where the initial selection in the app is based on the functional currency of the selected entity, but you want the user to be able to change the currency.

To accomplish this, follow these steps:

Note: Substitute CURRENCY for the name of your currency dimension, or use an attribute token or $CORE.DimensionInfo.KeyDimensionList[4]$ for CURRENCY for flexibility.

1. In the pre-selection add the following to support currency selection:

```
//Configure currency selections for re-select in view
Create GLOBALVARIABLE CURRENCYDefault AS STRING
Create GLOBALVARIABLE CURRENCYOptions AS STRING
Create GLOBALVARIABLE CURRENCYSymbols AS RANGE
Create GLOBALVARIABLE oldSelectDimensionList[] AS STRING //used to
restore symbol selection to original options

Set VARIABLE CURRENCYOptions = "11000"
Set VARIABLE CURRENCYSymbols = "SOURCEC###"

Create VARIABLE isCurrencySelection AS NUM //used to track in post-
selection if the is changing currency only
```

2. In the Post-Selection add logic to set the native currency on entity selection:

```
If Not(VariableExists("CURRENCY"))
    Create VARIABLE CURRENCY AS STRING
END If


If $isCurrencySelection$ //reset selection options to allow regular
symbol selections
    Set VARIABLE FORM_SelectDimensionList = ListAppend(CORE_EmptyList,
oldSelectDimensionList)
    Set VARIABLE isCurrencySelection = 0
Else
    Set VARIABLE CURRENCY = "[[SYMBOL,ZGPNativeCurrency,$ENTITIES$]]"
//apply currency of selected entity
END If
```

3. Add a View Action to allow the user to change currency (this will be a new button on the toolbar)

```
Set VARIABLE oldSelectDimensionList = ListAppend(CORE_EmptyList, FORM_
SelectDimensionList)
Set VARIABLE isCurrencySelection = 1


Set VARIABLE FORM_SelectDimensionList = ListAppend(CORE_EmptyList,
"CURRENCY")


Run PROCEDURE "VIEW\OnReselect.lvpro"
```

Specifying pre-export commands

Use pre-export commands to specify any commands to execute after the data area has been created and downloaded, but before the export has been executed.

Specify the data area definition

Use Data Area Definition to specify the data area to be queried for the export. Open the data area definition to specify its properties.

> **Note:** The name of the data area definition is Main.lvdsp and the name of the data area created using it is daMain.

1. For the Name, enter the name of your Trigger Area

> **Note:** The name must 29 or fewer characters and only contain alphanumeric characters and underscore "_".

2. In the Options table, complete the following fields:

Field	Notes
Data type	Specify the data to be queried using one of the following options: ■ ADJUSTED — Indicates that data queries include adjustments made via journal entries ■ UNADJUSTED — Indicates that data queries exclude adjustments made via journal entries Default is ADJUSTED.
Download type	Specify the type of data to be downloaded using one of the following options (download option specific in brackets): ■ Exclude parent values — Indicates that parent data is excluded in the download (STANDARDLEAFONLY) ■ Exclude parent and calculated values — Indicates that only leaf data that was submitted to the database via import, input or event calculation is included in the download (LEAFDATA). Default is All data.

3. In the Trigger Area table, complete the following fields for each dimension:

> **Note:** The Spec and Level fields are not applicable if a symbol contains a variable.

Field	Notes
Symbols	Type a symbol name. Alternatively, you can use the symbol selector. To add more than one symbol for a dimension, select a row containing the dimension for which you want to add another symbol and click Symbol. To move a symbol up or down within a dimension, click Move Up or Move Down. **Note:** To use the symbol selected by the user or defined by an attribute, enter the name of the dimension as a variable. For example, $ENTITIES$.

Field	Notes
Spec	Indicate the symbol specification using one of the following values: ■ All: Includes all symbols within the selected symbol's hierarchy. ■ Leaf: Includes only leaf symbols within the selected symbol's hierarchy. ■ Parent: Includes only parent symbols within the selected symbol's hierarchy. ■ Root and Parent: Includes only root and parent symbols within the selected symbol's hierarchy. **Note:** To use the symbol selected by the user or defined by an attribute, enter the name of the dimension as a variable. For example, $ENTITIES$.
Level	Specify the number of levels down from the symbol to be included in the data area.

4. Use Additional Configuration to specify any other Data Area Definition features you wish to use. Additional Configuration is a code editor that allows you to enter data area definition functions to be included in the data area.

   For more information on the code editor, see Using the Code Editor.

   Typical use of Additional Configuration in a data area definition is to

   - Add an attribute filter using the AttributeFilter function
   - Add a schedule to the using the Schedule function
   - Add a temporary symbol using the TempSym function

Specifying the data export spec

Use the data area export spec to specify how the data downloaded to the data area is converted to records in the target file.

To specify the data area export spec:

1. Open the Data Area Export Spec.
2. Fill in the Target File table:

Field	Notes
Target file	Specify the path and name of the data target. **Note:** If the user is selecting the file you can use the name of the variable here. In addition, you must set the property EXP.FileName for the app to execute correctly.

Field	Notes
Field Delimiter	Specify whether the character used to separate fields in the data source is the brace ({), comma (,), semicolon (;), pipe (|), or regional delimiter for comma separated values (, (regional)). Alternatively, you can type a field delimiter of your choice. The default is the regional delimiter for comma separated values (, (regional)).

3. The Dimension Definitions section is used to specify how each dimension is processed during export. For Dimension Definitions, complete the following:

   - Schedule: This field displays only if your system is configured to use schedules and you have access to schedules.

     a. The default of None indicates the export spec is for base data only.

     b. Specify the schedule to use for the import spec, and the schedule dimensions appear at the bottom of the list of Dimensions.

   - For each dimension, complete the following fields:

Field	Notes
Dimension	The name of the dimension.
Type	Specify the type of relationship between dimension symbols and field values in the data source using one of the following values: ■ Map — The dimension symbols are mapped to values as specified in the indicated map. ■ Match — The dimension symbols are written out with their names. ■ Unique — Values for the symbol specified are exported, but the symbol name is not written in the target file. **Note:** For a multiple values export spec, the dimension you indicated has Values for the Type field and cannot be changed.
Field Position	This field is available only when Type is Map or Match. Specify the field position to assign the dimension to using a positive integer. For example, for the Accounts dimension, type 3 if the account will be the third column of the data target.

Field	Notes
Symbol	This field is available only when Type is Unique. Type the symbol name for the specified dimension to which the data is imported. Alternatively, you can use the symbol selector. **Note:** You can also use a variable to define the symbol.
Map Location	This field is available only when Type is Map. Indicate the location of the map using one of the following values: ■ Internal — Indicates the map exists inside the database, created using the Mappings editor. For more information, see Maintaining mappings. ■ External — Indicates the map exists as a file, such as a text file on your local machine or network location. ■ Document — Indicates the map exists as a document within the Symbol Maps folder of the app.
Map	This field is available only when Type is Map. Do one of the following: ■ If you selected Internal as the Map Location, select a map from the list of available maps in the system. ■ If you selected External as the Map Location, type the path of the external map or click the ellipsis button (...) to select the external map. ■ If you selected Document as the Map Location, select a map from the list of available maps in the app. **Note:** You can use a variable for the Map field if the map location is Internal or External.

4. Fill in the Value Field Table:

Field	Notes
Value field name	Specify the name of the field that contains the value. The field name is used to specify filters. The default value for this field is 'value'.
Value field number	Specify the field position at which the data value will be written using a number.
v26.2 Value field decimals	Specify the number of decimal places for exported numeric values. This setting controls the precision of the exported numbers. If not specified, the default is 9 decimal places.

5. In the Data Options table, complete the following fields:

Field	Notes
Reverse value for credit accounts	Specify the path and name of the data target. Select this option to reverse the sign for numeric values for accounts with Credit as the Balance Type. For more information on the balance type in the ACCOUNTS dimension, see “Working with dimensions” in the *Longview Application Administrator Guide*. The default is No.
Decimal character	Specify the decimal character for the data import app using one of the following values: ■ “ . ” — The period is the decimal character. ■ “ , ” — The comma is the decimal character. ■ “ . (regional) ” — Use the regional character for the decimal. The default is the regional character for decimal “ . (regional) ”.
Allow duplicate mappings	This option applies only when at least one dimension’s Type is Map. Select this option to indicate that duplicate mappings are permitted. If multiple mappings are found, the system does the following for each type of mapping: ■ Exact — All duplicate Exact mappings are used so that the same value is imported to multiple data cells. If there are duplicate Exact and Wildcard or Range mappings, the Exact mapping is used, and the Wildcard or Range mapping is ignored. ■ Wildcard — If no Exact mappings are found, the value is imported to the first Wildcard or Range mapping listed. ■ Range — If no Exact mappings are found, the value is imported to the first Wildcard or Range mapping listed. If you do not select this option, duplicate mappings are not permitted. If multiple mappings are found, the system reports an error for each mapping. The default is No.

Field	Notes
Handling for duplicate records	Specify the way duplicate records are handled for the data import app using one of the following values: ■ Output all — Specifies that each duplicate record will be written to the target file. ■ Error — Specifies that duplicate records should not be allowed. The first entry is submitted. If a duplicate record is encountered, the system reports an error. ■ Sum values — Specifies that duplicate records should be allowed and the value of all such records summed. If a duplicate record is encountered that contains text (versus numeric) data, it is treated as an invalid/error record. The default is Error.
Number of errors before processing stops	Specify the maximum number of error records to permit before stopping the import process. The default is 0.
Custom header	When the header option is set to Custom, enter the header records in the form: 'header record', '...', 'header record'. Each header record is output on a new line in the target file.

6. You can optionally specify filters to apply when exporting data. For each filter type an expression using the following guidelines:

   - The expression can be joined by two or more expressions using AND or OR and grouped by brackets, as appropriate.
   - Each expression uses field names, operators, and values as follows: FieldName EQ|NE|LE|LT|GE|GT Value. If Value is a string, enclose the string in double quotation marks.
   - You can optionally include wildcard characters (? and *) and symbols for Value.

     Note: In this case do not enclose Value in double quotation marks.

   - You can optionally use symbols for operators, such as ==, !=, <=, <, >=, and >.
   - You must use Field# for FieldName to refer to a specific field on which you want to filter. For example, if Products is the fourth column in your target file and you want to filter Products to import only the Products_Default symbol, use Field4 == "Products_Default" as the expression.

Note: For more information on creating expressions, see "Using conditional operators" and "Search" in the *Longview Developer's Guide*.

Specifying post-export commands

Use post-export commands to specify any commands to execute after the export has been executed.

Specifying Symbol Maps

Symbol Maps(.lvmap) specify instructions when the symbol names in your data target do not match the symbol names in your Longview database.

For more information, see “Developing Longview ImportSpecs, ExportSpecs, and external Maps” in the *Longview Developer’s Guide*.

For more information on using the code editor, see Using the code editor.

Specifying forms

Use Forms to capture user input:

1. In an action within an input template
2. During post-selection in an app

For each form complete the following fields:

Field	Notes
Name	Enter the name of the form.
Title	Enter the title for the form. This is optional. If left blank the title of the form will be the same as the description of the app.
Width (in pixels)	Enter the width of the form. If not specified, the default width of 450 is used.
Height (in pixels)	Enter the height of the form. If not specified, the height is determined by the number of controls in the form.
Format for the date value returned	Enter the format a date is stored in an application framework variable when returned from a form. The format in the form is determined by the user’s regional settings.
Form controls	Add the controls to be displayed in the form via the code editor. For more information on the code editor, see Using the code editor.

Creating Multiple Value Data Area Export Apps

You can use Longview Designer to create multiple value data area export apps.

Working with variables

You can use Longview Designer to create, modify, reorder, or delete variables for your app. Variables are optional. You might include variables in your app if you want to, for example, use an attribute such as the current time period, or if you want to prompt a user to select a symbol or a file.

Once you have created variables, you can use them elsewhere in your app by enclosing them in dollar signs ($ $).

For more information, see “Variables” in the *Longview Developer’s Guide*.

You can also use external variables not defined on the Variables page of your app by enclosing them in dollar signs ($ $).

Variables tasks include:

- Creating variables
- Modifying variables
- Reordering variables
- Deleting variables

Creating variables

You can use Longview Designer to create variables for your app. Each variable you specify is created and set to the specified value when your app runs.

To create a variable:

1. Open the Variables section.
2. Do one of the following:
   - To create a new variable at the bottom of the list, click Variable. The new variable appears at the bottom of the variables list.
   - To create a new variable below a specific variable, select a row and click Variable. The new variable appears below the selected row.
3. Complete the following fields:

Field	Notes
Name	Type the name of the variable. Do not prefix the variable name with “LV_” or “LVS_”.
Type	Specify the type of data to be stored in the variable using one of the following options: ■ String — Designates a string value. ■ Range — Designates a list of symbols. ■ Number — Designates a numeric value. ■ List — Designates a list of string values. ■ NumberList — Designates a list of numeric values. The default is String.
Expression	Specify the value that the variable is set to. Based on the Type field this can be a number, a string, or a list of symbols, respectively.

Note: You can include attribute values in an expression. To include an attribute value in an expression, you must first set a variable to the value of the attribute using an attribute token, and then use the variable. For example, you could create a string variable named “currentPeriod” and use the expression “[[SYSTEM,SGPCurrentPeriod,DBDefault]]”. For more information on setting variables, see “Set Variable” in the *Longview Developer’s Guide*. For more information on using attributes, see “Attribute tokens” in the *Longview Developer’s Guide*.

Modifying variables

You can use Longview Designer to modify existing variables for your app.

To modify a variable:

1. Open the Variables section.
2. Select the variable you want to modify and make the necessary changes.

Reordering variables

You can use Longview Designer to change the order of variables in your app. The order of variables is important because you cannot use a variable in an expression until you first define the variable. Variables are created in the same order in which they are defined.

To reorder a variable:

1. Open the Variables section.
2. Select the variable you want to reorder, and then click either Move Up or Move Down. The variable moves one row in the indicated direction.
3. If you want to continue to move the selected variable, repeat step 2 until the variable is in the desired location.

Deleting variables

You can use Longview Designer to delete variables from your app.

To delete a variable:

1. Open the Variables section.
2. In the variables list, select the variable you want to delete, and click Delete. The variable is deleted.

Specifying pre-selection commands

Use pre-selection to perform any commands before symbol selection. The typical use of pre-selection is to perform advanced logic to determine default selections or set the range of possible symbols. Pre-selection commands are executed after symbol selection settings are applied from the app configuration, so all symbol selection variables have been created at this point.

See Generating a symbol selection form for details on symbol selection variables.

Note: Using pre-select commands to set default or symbols overrides the settings in User Selections.

Variable	Notes
<Dimension>Default	Specify the default selection in the symbol selector, for the Dimension.
<Dimension>Symbols	Specify the list of available selections in the symbol selector for the Dimension.

Example: Setting the list of possible symbols to all YTD periods

```
Create VARIABLE ytdPeriodList AS RANGE
Set VARIABLE ytdPeriodList = "[[SYSTEM,SGPTimePeriodsYTD,DBDEFAULT]]"
For EACH ytdPeriod in ytdPeriodList
    Set VARIABLE TIMEPERIODSSymbols = ListAppend(TIMEPERIODSSymbols,
"$ytdPeriod$#99")
Next
```

Specifying user selections

Use user selections to specify dimensions for which the user will be prompted to choose symbol(s). For each dimension that the user will choose symbols, right-click on the User Selections folder and choose Add Dimension. The dimension settings page will appear.

Variable	Notes
Dimension	Select the dimension the user will be prompted to select symbol(s) for.
Available selections	Select a symbol to limit the possible symbols available to the user. If left blank the user will be able to select from all symbols available in the dimension. The available symbols will always be limited by user access.
Default selection	Select a symbol to specify as the default selection. This can either be a database symbol, an attribute symbol, or a floating time period. **Note:** Attribute symbols and floating time periods will only be available if they are defined in your system. For more information, see “Specifying attributes in Longview Application Administrator” in the *Longview Analysis and Reporting Guide*.
Selection required	Check this to force the user to select a symbol to continue.
Allow leaf selection	Check this to allow the user to select a leaf symbol.
Allow parent selection	Check this to allow the user to select a parent symbol.
Allow read-only selection	Check this to allow the user to select a read-only symbol.

Variable	Notes
Allow multiple selections	Check this to allow the user to select more than one symbol.
Attribute filter	Enter an attribute filter expression to further limit the possible symbol selections. For more information see SymbolSelector in the *Longview Developer's Guide*.

Specifying attribute-based selections

Use Attribute Based Selections to specify selections for dimensions that are deter- mined by an attribute value. For each dimension to be determined via an attribute, right-click on Attribute Based Selections and select Add Dimension.

Variable	Notes
Dimension	Select the dimension to select symbol(s) using an attribute value.
Attribute class	Select the class of attribute to use.
Attribute name	Select the attribute.
Source dimension (SYMBOL class)	If the attribute class is SYMBOL, select the source dimension. **Note:** The source dimension must be a dimension specified as a user selection or attribute-based selection.

Note: The variable created to hold the attribute-based symbol will have the same name as the dimension. The variable created will be a STRING type.

Example, to set the currency selection to be the functional currency of the selected entity:

1. Select the **CURRENCIES** dimension.
2. Set the attribute class to **SYMBOL**.
3. Enter "ZGPNativeCurrency" for the attribute name.
4. Select **ENTITIES** for the source dimension.

Example, to set the time period selection to be the current period:

1. Select the **TIMEPERIODS** dimension.
2. Set the attribute class to **SYSTEM**.
3. Enter "SGPCurrentPeriod" for the attribute name.

Specifying post-selection commands

Use post-selection to perform any commands after symbol selection, but before any data areas are created. The typical use of post-selection commands is to perform advanced symbol selection tasks. For example, you might use post-selection to create a list of all standard intercompany accounts. In addition, you may want to use a form to further refine selection, or instead of the template provided symbol selection.

Caution: Post-selection commands are executed as part of symbol selection within a view. Any Create VARIABLE commands will cause an error during symbol reselection. If “Allow user to change selections in view” is allowed, either create variables only in pre-selection, or wrap variable creation in VariableExists checks.

Example: Get a list of all standard intercompany accounts

```
If Not(VariableExists("ACCOUNTS"))
    Create VARIABLE ACCOUNTS AS RANGE
END If
Set VARIABLE ACCOUNTS = CreateList(SYMBOLS, DATABASE, ACCOUNTS, "TB###")
Set VARIABLE ACCOUNTS = FilterList(ACCOUNTS, "
[[SYMBOL,ZElimICSchTransactions,THIS]] == 'ICStandard'")
```

Example: Specifying multiple attribute time periods (current forecast and budget period)

```
If Not(VariableExists("TIMEPERIODS"))
    Create VARIABLE TIMEPERIODS AS RANGE
END If
Set VARIABLE TIMEPERIODS = "[[SYSTEM,SGPForecastPeriod,DBDEFAULT]]" Set
VARIABLE TIMEPERIODS = ListAppend(TIMEPERIODS, "
[[SYSTEM,SGPBudgetPeriod,DBDEFAULT]]")
```

Example: Specify temporary symbols mixed with real symbols

```
If Not(VariableExists("TIMEPERIODS"))
    Create VARIABLE TIMEPERIODS AS RANGE
END If
Set VARIABLE TIMEPERIODS = "[[SYSTEM,SGPForecastPeriod,DBDEFAULT]]"
Set VARIABLE TIMEPERIODS = ListAppend(TIMEPERIODS, DIFF")
Set VARIABLE TIMEPERIODS = ListAppend(TIMEPERIODS, "
[[SYSTEM,SGPBudgetPeriod,DBDEFAULT]]")
```

To capture additional user input via a form, it is required that the button clicked be copied to the FORM_ ButtonClicked variable to ensure expected app execution. Specifically, app execution continues if FORM_ButtonClicked has a value of “OK”.

Example: Capture additional user input via a form

```
Show FORM USING Custom.lvfrm
Set VARIABLE FORM_ButtonClicked = $LVS_BUTTONCLICKED$
```

Allowing currency re-selection in a data view input app

There may be cases where the initial selection in the app is based on the functional currency of the selected entity, but you want the user to be able to change the currency.

To accomplish this, follow these steps:

Note: Substitute CURRENCY for the name of your currency dimension, or use an attribute token or $CORE.DimensionInfo.KeyDimensionList[4]$ for CURRENCY for flexibility.

1. In the pre-selection add the following to support currency selection:

```
//Configure currency selections for re-select in view
Create GLOBALVARIABLE CURRENCYDefault AS STRING
Create GLOBALVARIABLE CURRENCYOptions AS STRING
Create GLOBALVARIABLE CURRENCYSymbols AS RANGE
Create GLOBALVARIABLE oldSelectDimensionList[] AS STRING //used to
restore symbol selection to original options

Set VARIABLE CURRENCYOptions = "11000"
Set VARIABLE CURRENCYSymbols = "SOURCEC###"

Create VARIABLE isCurrencySelection AS NUM //used to track in post-
selection if the is changing currency only
```

2. In the Post-Selection add logic to set the native currency on entity selection:

```
If Not(VariableExists("CURRENCY"))
    Create VARIABLE CURRENCY AS STRING
END If

If $isCurrencySelection$ //reset selection options to allow regular
symbol selections
    Set VARIABLE FORM_SelectDimensionList = ListAppend(CORE_EmptyList,
oldSelectDimensionList)
    Set VARIABLE isCurrencySelection = 0
Else
    Set VARIABLE CURRENCY = "[[SYMBOL,ZGPNativeCurrency,$ENTITIES$]]"
//apply currency of selected entity
END If
```

3. Add a View Action to allow the user to change currency (this will be a new button on the toolbar)

```
Set VARIABLE oldSelectDimensionList = ListAppend(CORE_EmptyList, FORM_
SelectDimensionList)
Set VARIABLE isCurrencySelection = 1


Set VARIABLE FORM_SelectDimensionList = ListAppend(CORE_EmptyList,
"CURRENCY")


Run PROCEDURE "VIEW\OnReselect.lvpro"
```

Specifying pre-export commands

Use pre-export commands to specify any commands to execute after the data area has been created and downloaded, but before the export has been executed.

Specify the data area definition

Use Data Area Definition to specify the data area to be queried for the export. Open the data area definition to specify its properties.

Note: The name of the data area definition is Main.lvdsp and the name of the data area created using it is daMain.

1. For the Name, enter the name of your Trigger Area

   Note: The name must 29 or fewer characters and only contain alphanumeric characters and underscore “_”.

2. In the Options table, complete the following fields:

Field	Notes
Data type	Specify the data to be queried using one of the following options: ■ ADJUSTED — Indicates that data queries include adjustments made via journal entries ■ UNADJUSTED — Indicates that data queries exclude adjustments made via journal entries Default is ADJUSTED.

Field	Notes
Download type	Specify the type of data to be downloaded using one of the following options (download option specific in brackets): ■ Exclude parent values — Indicates that parent data is excluded in the download (STANDARDLEAFONLY) ■ Exclude parent and calculated values — Indicates that only leaf data that was submitted to the database via import, input or event calculation is included in the download (LEAFDATA). Default is All data.

3. In the Trigger Area table, complete the following fields for each dimension:

Note: The Spec and Level fields are not applicable if a symbol contains a variable.

Field	Notes
Symbols	Type a symbol name. Alternatively, you can use the symbol selector. To add more than one symbol for a dimension, select a row containing the dimension for which you want to add another symbol and click Symbol. To move a symbol up or down within a dimension, click Move Up or Move Down. **Note:** To use the symbol selected by the user or defined by an attribute, enter the name of the dimension as a variable. For example, $ENTITIES$.
Spec	Indicate the symbol specification using one of the following values: ■ All: Includes all symbols within the selected symbol's hierarchy. ■ Leaf: Includes only leaf symbols within the selected symbol's hierarchy. ■ Parent: Includes only parent symbols within the selected symbol's hierarchy. ■ Root and Parent: Includes only root and parent symbols within the selected symbol's hierarchy. **Note:** To use the symbol selected by the user or defined by an attribute, enter the name of the dimension as a variable. For example, $ENTITIES$.
Level	Specify the number of levels down from the symbol to be included in the data area.

4. Use Additional Configuration to specify any other Data Area Definition features you wish to use. Additional Configuration is a code editor that allows you to enter data area definition functions to be included in the data area.

   For more information on the code editor, see Using the Code Editor.

   Typical use of Additional Configuration in a data area definition is to

   - Add an attribute filter using the AttributeFilter function
   - Add a schedule to the using the Schedule function
   - Add a temporary symbol using the TempSym function

Specifying the data export spec

Use the data area export spec to specify how the data downloaded to the data area is converted to records in the target file.

To specify the data area export spec:

1. Open the Data Area Export Spec.
2. Fill in the Target File table:

Field	Notes
Target file	Specify the path and name of the data target. **Note:** If the user is selecting the file you can use the name of the variable here. In addition, you must set the property EXP.FileName for the app to execute correctly.
Field Delimiter	Specify whether the character used to separate fields in the data source is the brace ({), comma (,), semicolon (;), pipe (|), or regional delimiter for comma separated values (, (regional)). Alternatively, you can type a field delimiter of your choice. The default is the regional delimiter for comma separated values (, (regional)).
Dimension corresponding to values	Specify the dimension for which there are multiple value fields.

3. In the Data Options table, complete the following fields:

Field	Notes
Reverse value for credit accounts	Specify the path and name of the data target. Select this option to reverse the sign for numeric values for accounts with Credit as the Balance Type. For more information on the balance type in the ACCOUNTS dimension, see “Working with dimensions” in the *Longview Application Administrator Guide*. The default is No.

Field	Notes
Decimal character	Specify the decimal character for the data import app using one of the following values: ■ " . " — The period is the decimal character. ■ " , " — The comma is the decimal character. ■ " . (regional) " — Use the regional character for the decimal. The default is the regional character for decimal " . (regional) ".
Allow duplicate mappings	This option applies only when at least one dimension's Type is Map. Select this option to indicate that duplicate mappings are permitted. If multiple mappings are found, the system does the following for each type of mapping: ■ Exact – All duplicate Exact mappings are used so that the same value is imported to multiple data cells. If there are duplicate Exact and Wildcard or Range mappings, the Exact mapping is used, and the Wildcard or Range mapping is ignored. ■ Wildcard – If no Exact mappings are found, the value is imported to the first Wildcard or Range mapping listed. ■ Range – If no Exact mappings are found, the value is imported to the first Wildcard or Range mapping listed. If you do not select this option, duplicate mappings are not permitted. If multiple mappings are found, the system reports an error for each mapping. The default is No.
Handling for duplicate records	Specify the way duplicate records are handled for the data import app using one of the following values: ■ Output all — Specifies that each duplicate record will be written to the target file. ■ Error — Specifies that duplicate records should not be allowed. The first entry is submitted. If a duplicate record is encountered, the system reports an error. ■ Sum values — Specifies that duplicate records should be allowed and the value of all such records summed. If a duplicate record is encountered that contains text (versus numeric) data, it is treated as an invalid/error record. The default is Error.
Number of errors before processing stops	Specify the maximum number of error records to permit before stopping the import process. The default is 0.

Field	Notes
Custom header	When the header option is set to Custom, enter the header records in the form: 'header record', '...', 'header record'. Each header record is output on a new line in the target file.

4. The Dimension Definitions section is used to specify how each dimension is processed during export. For Dimension Definitions, complete the following:

   - Schedule: This field displays only if your system is configured to use schedules and you have access to schedules.
   - The default of None indicates the export spec is for base data only.
   - Specify the schedule to use for the import spec, and the schedule dimensions appear at the bottom of the list of Dimensions.
   - For each dimension, complete the following fields:

Field	Notes
Dimension	The name of the dimension.
Type	Specify the type of relationship between dimension symbols and field values in the data source using one of the following values: • Map — The dimension symbols are mapped to values as specified in the indicated map. • Match — The dimension symbols are written out with their names. • Unique — Values for the symbol specified are exported, but the symbol name is not written in the target file. **Note:** For a multiple values export spec, the dimension you indicated has Values for the Type field and cannot be changed.
Field Name	This field is available only when Type is Map or Match. Optionally, enter the name of the field. If the field name is not specified it will be set to Field with the Field Position appended. The field name is used to specify filters.
Field Position	This field is available only when Type is Map or Match. Specify the field position to assign the dimension to using a positive integer. For example, for the Accounts dimension, type 3 if the account will be the third column of the data target.

Field	Notes
Symbol	This field is available only when Type is Unique. Type the symbol name for the specified dimension to which the data is imported. Alternatively, you can use the symbol selector. **Note:** You can also use a variable to define the symbol.
Map Location	This field is available only when Type is Map. Indicate the location of the map using one of the following values: • Internal — Indicates the map exists inside the database, created using the Mappings editor. For more information, see "Maintaining mappings". • External — Indicates the map exists as a file, such as a text file on your local machine or network location. • Document — Indicates the map exists as a document within the Symbol Maps folder of the app.
Map	This field is available only when Type is Map. Do one of the following: • If you selected Internal as the Map Location, select a map from the list of available maps in the system. • If you selected External as the Map Location, type the path of the external map or click the ellipsis button (...) to select the external map. • If you selected Document as the Map Location, select a map from the list of available maps in the app. **Note:** You can use a variable for the Map field if the map location is Internal or External.

5. The Values Field section is used to specify the field position for each symbol to be exported in the Values dimension. Complete the following fields:

Field	Notes
Field Name	Optionally, enter the name of the field. If the field name is not specified it will be set to Field with the Field Position appended. The field name is used to specify filters.
Field Position	Specify the field position of the value field using a positive integer, where the field position is the corresponding column in the data source containing the data. For example, if the value is exported to the third column in the target file, type 3.

Field	Notes
v26.2 Decimal Places	Select the number of decimal places to apply to the exported value. Choose a value from 0-9 or leave the field blank. If not specified, existing exports default to 9. The selected value determines the decimal precision used in the export.
Symbol	Type the name of a symbol from the dimension indicated in Multiple Value Fields for Dimension for the value in the corresponding field position. Alternatively, you can use the symbol selector. **Note:** You can use a variable for the Symbol field.

Note: You can delete a value field by selecting a value field row and clicking Delete.

6. You can optionally specify filters to apply when exporting data. For each filter type an expression using the following guidelines:
   - The expression can be joined by two or more expressions using AND or OR and grouped by brackets, as appropriate.
   - Each expression uses field names, operators, and values as follows: FieldName EQ|NE|LE|LT|GE|GT Value. If Value is a string, enclose the string in double quotation marks.
   - You can optionally include wildcard characters (? and *) and symbols for Value.

     Note: In this case do not enclose Value in double quotation marks.
   - You can optionally use symbols for operators, such as ==, !=, <=, <, >=, and >.
   - You must use Field# for FieldName to refer to a specific field on which you want to filter. For example, if Products is the fourth column in your target file and you want to filter Products to import only the Products_Default symbol, use Field4 == "Products_Default" as the expression.

Note: For more information on creating expressions, see “Using conditional operators” and “Search” in the *Longview Developer’s Guide*.

Specifying post-export commands

Use post-export commands to specify any commands to execute after the export has been executed.

Specifying Symbol Maps

Symbol Maps(.lvmap) specify instructions when the symbol names in your data target do not match the symbol names in your Longview database.

For more information, see “Developing Longview ImportSpecs, ExportSpecs, and external Maps” in the *Longview Developer’s Guide*.

For more information on using the code editor, see Using the code editor.

Specifying forms

Use Forms to capture user input:

1. In an action within an input template
2. During post-selection in an app

For each form complete the following fields:

Field	Notes
Name	Enter the name of the form.
Title	Enter the title for the form. This is optional. If left blank the title of the form will be the same as the description of the app.
Width (in pixels)	Enter the width of the form. If not specified, the default width of 450 is used.
Height (in pixels)	Enter the height of the form. If not specified, the height is determined by the number of controls in the form.
Format for the date value returned	Enter the format a date is stored in an application framework variable when returned from a form. The format in the form is determined by the user's regional settings.
Form controls	Add the controls to be displayed in the form via the code editor. For more information on the code editor, see Using the code editor.

Creating Attribute-Based Report Apps

You can use attribute-based report apps to analyze your Longview data based on defined attribute values.

Specifying overall settings

You can specify overall display and formatting options for the report.

Display options:

Field	Notes
Display dimensions	Select the dimensions that you would like to display in the attribute-based report. **Note:** The Display Dimension cannot be a Dimension corresponding to values.

Field	Notes
Output	Select the way you would like to view your display dimensions: ■ Name: A single column for each Display dimension will be visible showing the symbols name ■ Description: A single column for each Display dimension will be visible showing the symbols description. ■ Name and Description: Two columns for each display dimension will be visible - one displaying the symbols name and the other displaying the symbols description.

Values options:

Field	Notes
Dimension corresponding to values	Specify the dimension for which you would like to see the values. **Note:** The Dimension corresponding to values cannot be a Display Dimension.
Output	Select the way the column header(s) will display for a values dimension: ■ Name: A single column for each Display dimension will be visible showing the symbols name ■ Description: A single column for each Display dimension will be visible showing the symbols description. ■ Name and Description: Two columns for each display dimension will be visible - one displaying the symbols name and the other displaying the symbols description.
Data type	Select ADJUSTED to query adjusted data (unadjusted + journal entries) or UNADJUSTED to query data excluding journal entries

Number formatting:

Field	Notes
Default decimals	Enter the number of decimals to display for each numeric value. Generally this should be from 0 to 9.
Use parentheses for negative numbers	Select Yes to display negative numbers with parentheses. Select No to display negative numbers with a negative sign
Use thousands separator for numbers	Select Yes to display a locale-specific thousands separator based on the operating system setting. Select No to not display a thousands separator.

Specifying attribute columns

Use the Attribute Columns to add attributes you would like to analyze to your attribute-based report.

To add an attribute column

1. Right-click on attribute **Columns** and select **Add Attribute**.
2. Fill in the following fields:
    a. Attribute: Select the **symbol attribute** you would like to display in the report.
    b. Description: Specify the description that will display as the column header for the attribute column.
    c. Source dimension: Specify the source dimension the symbol attribute data is based on.

Specifying pre-selection commands

Use pre-selection to perform any commands before symbol selection. The typical use of pre-selection is to perform advanced logic to determine default selections or set the range of possible symbols. Pre-selection commands are executed after symbol selection settings are applied from the app configuration, so all symbol selection variables have been created at this point.

See Generating a symbol selection form for details on symbol selection variables.

Note: Using pre-select commands to set default or symbols overrides the settings in User Selections.

Variable	Notes
<Dimension>Default	Specify the default selection in the symbol selector, for the Dimension.
<Dimension>Symbols	Specify the list of available selections in the symbol selector for the Dimension.

Example: Setting the list of possible symbols to all YTD periods

```
Create VARIABLE ytdPeriodList AS RANGE
Set VARIABLE ytdPeriodList = "[[SYSTEM,SGPTimePeriodsYTD,DBDEFAULT]]"
For EACH ytdPeriod in ytdPeriodList
    Set VARIABLE TIMEPERIODSSymbols = ListAppend(TIMEPERIODSSymbols,
"$ytdPeriod$#99")
Next
```

Specifying user selections

Use user selections to specify dimensions for which the user will be prompted to choose symbol(s). For each dimension that the user will choose symbols, right-click on the User Selections folder and choose Add Dimension. The dimension settings page will appear.

Variable	Notes
Dimension	Select the dimension the user will be prompted to select symbol(s) for.
Available selections	Select a symbol to limit the possible symbols available to the user. If left blank the user will be able to select from all symbols available in the dimension. The available symbols will always be limited by user access.
Default selection	Select a symbol to specify as the default selection. This can either be a database symbol, an attribute symbol, or a floating time period. **Note:** Attribute symbols and floating time periods will only be available if they are defined in your system. For more information, see “Specifying attributes in Longview Application Administrator” in the *Longview Analysis and Reporting Guide*.
Selection required	Check this to force the user to select a symbol to continue.
Allow leaf selection	Check this to allow the user to select a leaf symbol.
Allow parent selection	Check this to allow the user to select a parent symbol.
Allow read-only selection	Check this to allow the user to select a read-only symbol.
Allow multiple selections	Check this to allow the user to select more than one symbol.
Attribute filter	Enter an attribute filter expression to further limit the possible symbol selections. For more information see SymbolSelector in the *Longview Developer’s Guide*.

Specifying attribute-based selections

Use Attribute Based Selections to specify selections for dimensions that are deter- mined by an attribute value. For each dimension to be determined via an attribute, right-click on Attribute Based Selections and select Add Dimension.

Variable	Notes
Dimension	Select the dimension to select symbol(s) using an attribute value.
Attribute class	Select the class of attribute to use.
Attribute name	Select the attribute.
Source dimension (SYMBOL class)	If the attribute class is SYMBOL, select the source dimension. **Note:** The source dimension must be a dimension specified as a user selection or attribute-based selection.

Note: The variable created to hold the attribute-based symbol will have the same name as the dimension. The variable created will be a STRING type.

Example, to set the currency selection to be the functional currency of the selected entity:

1. Select the **CURRENCIES** dimension.
2. Set the attribute class to **SYMBOL**.
3. Enter “ZGPNativeCurrency” for the attribute name.
4. Select **ENTITIES** for the source dimension.

Example, to set the time period selection to be the current period:

1. Select the **TIMEPERIODS** dimension.
2. Set the attribute class to **SYSTEM**.
3. Enter “SGPCurrentPeriod” for the attribute name.

Specifying post-selection commands

Use post-selection to perform any commands after symbol selection, but before any data areas are created. The typical use of post-selection commands is to perform advanced symbol selection tasks. For example, you might use post-selection to create a list of all standard intercompany accounts. In addition, you may want to use a form to further refine selection, or instead of the template provided symbol selection.

Caution: Post-selection commands are executed as part of symbol selection within a view. Any Create VARIABLE commands will cause an error during symbol reselection. If “Allow user to change selections in view” is allowed, either create variables only in pre-selection, or wrap variable creation in VariableExists checks.

Example: Get a list of all standard intercompany accounts

```
If Not(VariableExists("ACCOUNTS"))
    Create VARIABLE ACCOUNTS AS RANGE
END If
Set VARIABLE ACCOUNTS = CreateList(SYMBOLS, DATABASE, ACCOUNTS, "TB###")
Set VARIABLE ACCOUNTS = FilterList(ACCOUNTS, "
[[SYMBOL,ZElimICSchTransactions,THIS]] == 'ICStandard'")
```

Example: Specifying multiple attribute time periods (current forecast and budget period)

```
If Not(VariableExists("TIMEPERIODS"))
    Create VARIABLE TIMEPERIODS AS RANGE
END If
```

```
Set VARIABLE TIMEPERIODS = "[[SYSTEM,SGPForecastPeriod,DBDEFAULT]]" Set
VARIABLE TIMEPERIODS = ListAppend(TIMEPERIODS, "
[[SYSTEM,SGPBudgetPeriod,DBDEFAULT]]")
```

Example: Specify temporary symbols mixed with real symbols

```
If Not(VariableExists("TIMEPERIODS"))
    Create VARIABLE TIMEPERIODS AS RANGE
END If
Set VARIABLE TIMEPERIODS = "[[SYSTEM,SGPForecastPeriod,DBDEFAULT]]"
Set VARIABLE TIMEPERIODS = ListAppend(TIMEPERIODS, DIFF")
Set VARIABLE TIMEPERIODS = ListAppend(TIMEPERIODS, "
[[SYSTEM,SGPBudgetPeriod,DBDEFAULT]]")
```

To capture additional user input via a form, it is required that the button clicked be copied to the FORM_ ButtonClicked variable to ensure expected app execution. Specifically, app execution continues if FORM_ButtonClicked has a value of “OK”.

Example: Capture additional user input via a form

```
Show FORM USING Custom.lvfrm
Set VARIABLE FORM_ButtonClicked = $LVS_BUTTONCLICKED$
```

Allowing currency re-selection in a data view input app

There may be cases where the initial selection in the app is based on the functional currency of the selected entity, but you want the user to be able to change the currency.

To accomplish this, follow these steps:

Note: Substitute CURRENCY for the name of your currency dimension, or use an attribute token or $CORE.DimensionInfo.KeyDimensionList[4]$ for CURRENCY for flexibility.

1. In the pre-selection add the following to support currency selection:

```
//Configure currency selections for re-select in view
Create GLOBALVARIABLE CURRENCYDefault AS STRING
Create GLOBALVARIABLE CURRENCYOptions AS STRING
Create GLOBALVARIABLE CURRENCYSymbols AS RANGE
Create GLOBALVARIABLE oldSelectDimensionList[] AS STRING //used to
restore symbol selection to original options


Set VARIABLE CURRENCYOptions = "11000"
```

```
Set VARIABLE CURRENCYSymbols = "SOURCEC###"


Create VARIABLE isCurrencySelection AS NUM //used to track in post-
selection if the is changing currency only
```

2. In the Post-Selection add logic to set the native currency on entity selection:

```
If Not(VariableExists("CURRENCY"))
    Create VARIABLE CURRENCY AS STRING
END If


If $isCurrencySelection$ //reset selection options to allow regular
symbol selections
    Set VARIABLE FORM_SelectDimensionList = ListAppend(CORE_EmptyList,
oldSelectDimensionList)
    Set VARIABLE isCurrencySelection = 0
Else
    Set VARIABLE CURRENCY = "[[SYMBOL,ZGPNativeCurrency,$ENTITIES$]]"
//apply currency of selected entity
END If
```

3. Add a View Action to allow the user to change currency (this will be a new button on the toolbar)

```
Set VARIABLE oldSelectDimensionList = ListAppend(CORE_EmptyList, FORM_
SelectDimensionList)
Set VARIABLE isCurrencySelection = 1


Set VARIABLE FORM_SelectDimensionList = ListAppend(CORE_EmptyList,
"CURRENCY")


Run PROCEDURE "VIEW\OnReselect.lvpro"
```

Specifying the data area definition

Use Data Area Definitions to specify the data areas to be used in the data input view. For each data area definition to be created, right-click on Data Area Definitions and select New Document.

Note: Only data areas that contain numeric data are supported for attribute-based reports.

1. In the Data Area table, complete the following fields for each dimension:

   Note: The Spec and Level fields are not applicable if a symbol contains a variable.

Field	Notes
Symbols	Type a symbol name. Alternatively, you can use the symbol selector. To add more than one symbol for a dimension, select a row containing the dimension for which you want to add another symbol and click Symbol. To move a symbol up or down within a dimension, click Move Up or Move Down. **Note:** To use the symbol selected by the user or defined by an attribute, enter the name of the dimension as a variable, for example $ENTITIES$
Spec	Indicate the symbol specification using one of the following values: ■ All: Includes all symbols within the selected symbol's hierarchy. ■ Leaf: Includes only leaf symbols within the selected symbol's hierarchy. ■ Parent: Includes only parent symbols within the selected symbol's hierarchy. ■ Root and Parent: Includes only root and parent symbols within the selected symbol's hierarchy. **Note:** To use the symbol selected by the user or defined by an attribute, enter the name of the dimension as a variable, for example $ENTITIES$
Level	Specify the number of levels down from the symbol to be included in the data area.

2. Use Additional Configuration to specify any other Data Area Definition features you wish to use. Additional Configuration is a code editor that allows you to enter data area definition functions to be included in the data area. For more information on the code editor, see Using the code editor.

   Typical use of Additional Configuration in a data area definition would be to:

   - Add an attribute filter using the AttributeFilter function
   - Add a schedule to the using the Schedule function
   - Add a temporary symbol using the TempSym function
   - Override the query type specified in Settings using the AdjustedDetail function

Creating Standard Intercompany Report Apps

You can use Longview Designer to create standard intercompany report apps.

Specifying settings

You can use Longview Designer to specify settings for a standard intercompany report app.

Note: For a matching report, each account in group 1 is matched to the account listed in the same order in group 2. It is important to list the accounts in the correct order for the matching to work. For example, in an intercompany accounts receivable report list in group 1: intercompany receivable followed by intercompany payable and in group 2: intercompany payable followed by intercompany receivable.

Field	Notes
Report type	Select the report type: ■ MATCHING: Best used when intercompany accounts have a 1 to 1 relationship (Example one receivable account for each payable account) **Note:** A user running this report will see data entered for entities the user has access to and the corresponding data entered by any contra entities. For example, running this report for intercompany receivables will show the related payable amount entered by the contra entity. ■ ELIMINATION: Used to view the elimination results by elimination entity **Note:** A user running this report must have read access to at least one elimination entity to see data in the report. ■ GROUPING: Best used when there are is not a 1 to 1 relationship between intercompany accounts **Note:** A user running this report will only see data entered for entities the user has access to. The contra entity will be reported without data.
Display group 1 accounts in header?	Select Yes to display the group 1 accounts selected in the header of the report.
Display group 2 accounts in header?	Select Yes to display the group 2 accounts selected in the header of the report.
Group 1 Accounts	Enter the group 1 accounts. Each account is separated by the pipe character (|). You can also select the symbols to include using the symbol specifier. Click the Search icon to use the symbol specifier.

Field	Notes
Group 2 Accounts	For MATCHING reports, enter the group 2 accounts. Each account is separated by the pipe character (|). You can also select the symbols to include using the symbol specifier. Click the Search icon to use the symbol specifier. **Note:** Leave this field blank for ELIMINATION and GROUPING report types.

Using the symbol specifier

The symbol specifier allows you to specify each account restriction using the symbol selector.

To specify symbols:

1. Click the Symbol button to add additional accounts.
2. For each row enter or select the account to include.

Note: A parent account can be selected, which will result in all accounts under it being included in the report.

Creating investment intercompany report apps

You can use Longview Designer to create investment intercompany report apps.

Specifying settings

You can use Longview Designer to specify settings for an investment intercompany report app.

Field	Notes
Display capital accounts in header?	Select Yes to display the capital accounts selected in the header of the report.
Capital accounts	Enter the capital accounts. Each account is separated by the pipe character (|). You can also select the symbols to include using the symbol specifier. Click the Search icon to use the symbol specifier.

Using the symbol specifier

The symbol specifier allows you to specify each account restriction using the symbol selector.

To specify symbols:

1. Click the **Symbol** button to add additional accounts.
2. For each row, enter or select the account to include.

Note: A parent account can be selected, which will result in all accounts under it being included in the report.

Creating Validation Reports

You can use Longview Designer to create, duplicate, modify, verify and delete validation report apps. You can use validation report apps to review validation data in the consolidation system.

Specifying pre-selection commands

Use pre-selection to perform any commands before symbol selection. The typical use of pre-selection is to perform advanced logic to determine default selections or set the range of possible symbols. Pre-selection commands are executed after symbol selection settings are applied from the app configuration, so all symbol selection variables have been created at this point.

See Generating a symbol selection form for details on symbol selection variables.

Note: Using pre-select commands to set default or symbols overrides the settings in User Selections.

Variable	Notes
<Dimension>Default	Specify the default selection in the symbol selector, for the Dimension.
<Dimension>Symbols	Specify the list of available selections in the symbol selector for the Dimension.

Example: Setting the list of possible symbols to all YTD periods

```
Create VARIABLE ytdPeriodList AS RANGE
Set VARIABLE ytdPeriodList = "[[SYSTEM,SGPTimePeriodsYTD,DBDEFAULT]]"
For EACH ytdPeriod in ytdPeriodList
    Set VARIABLE TIMEPERIODSSymbols = ListAppend(TIMEPERIODSSymbols,
"$ytdPeriod$#99")
Next
```

Specifying user selections

Use user selections to specify dimensions for which the user will be prompted to choose symbol(s). For each dimension that the user will choose symbols, right-click on the User Selections folder and choose Add Dimension. The dimension settings page will appear.

Variable	Notes
Dimension	Select the dimension the user will be prompted to select symbol(s) for.

Variable	Notes
Available selections	Select a symbol to limit the possible symbols available to the user. If left blank the user will be able to select from all symbols available in the dimension. The available symbols will always be limited by user access.
Default selection	Select a symbol to specify as the default selection. This can either be a database symbol, an attribute symbol, or a floating time period. **Note:** Attribute symbols and floating time periods will only be available if they are defined in your system. For more information, see “Specifying attributes in Longview Application Administrator” in the *Longview Analysis and Reporting Guide*.
Selection required	Check this to force the user to select a symbol to continue.
Allow leaf selection	Check this to allow the user to select a leaf symbol.
Allow parent selection	Check this to allow the user to select a parent symbol.
Allow read-only selection	Check this to allow the user to select a read-only symbol.
Allow multiple selections	Check this to allow the user to select more than one symbol.
Attribute filter	Enter an attribute filter expression to further limit the possible symbol selections. For more information see SymbolSelector in the *Longview Developer’s Guide*.

Specifying attribute-based selections

Use Attribute Based Selections to specify selections for dimensions that are deter- mined by an attribute value. For each dimension to be determined via an attribute, right-click on Attribute Based Selections and select Add Dimension.

Variable	Notes
Dimension	Select the dimension to select symbol(s) using an attribute value.
Attribute class	Select the class of attribute to use.
Attribute name	Select the attribute.
Source dimension (SYMBOL class)	If the attribute class is SYMBOL, select the source dimension. **Note:** The source dimension must be a dimension specified as a user selection or attribute-based selection.

Note: The variable created to hold the attribute-based symbol will have the same name as the dimension. The variable created will be a STRING type.

Example, to set the currency selection to be the functional currency of the selected entity:

1. Select the **CURRENCIES** dimension.
2. Set the attribute class to **SYMBOL**.
3. Enter “ZGPNativeCurrency” for the attribute name.
4. Select **ENTITIES** for the source dimension.

Example, to set the time period selection to be the current period:

1. Select the **TIMEPERIODS** dimension.
2. Set the attribute class to **SYSTEM**.
3. Enter “SGPCurrentPeriod” for the attribute name.

Specifying post-selection commands

Use post-selection to perform any commands after symbol selection, but before any data areas are created. The typical use of post-selection commands is to perform advanced symbol selection tasks. For example, you might use post-selection to create a list of all standard intercompany accounts. In addition, you may want to use a form to further refine selection, or instead of the template provided symbol selection.

Caution: Post-selection commands are executed as part of symbol selection within a view. Any Create VARIABLE commands will cause an error during symbol reselection. If “Allow user to change selections in view” is allowed, either create variables only in pre-selection, or wrap variable creation in VariableExists checks.

Example: Get a list of all standard intercompany accounts

```
If Not(VariableExists("ACCOUNTS"))
    Create VARIABLE ACCOUNTS AS RANGE
END If
Set VARIABLE ACCOUNTS = CreateList(SYMBOLS, DATABASE, ACCOUNTS, "TB###")
Set VARIABLE ACCOUNTS = FilterList(ACCOUNTS, "
[[SYMBOL,ZElimICSchTransactions,THIS]] == 'ICStandard'")
```

Example: Specifying multiple attribute time periods (current forecast and budget period)

```
If Not(VariableExists("TIMEPERIODS"))
    Create VARIABLE TIMEPERIODS AS RANGE
END If
```

```
Set VARIABLE TIMEPERIODS = "[[SYSTEM,SGPForecastPeriod,DBDEFAULT]]" Set
VARIABLE TIMEPERIODS = ListAppend(TIMEPERIODS, "
[[SYSTEM,SGPBudgetPeriod,DBDEFAULT]]")
```

Example: Specify temporary symbols mixed with real symbols

```
If Not(VariableExists("TIMEPERIODS"))
    Create VARIABLE TIMEPERIODS AS RANGE
END If
Set VARIABLE TIMEPERIODS = "[[SYSTEM,SGPForecastPeriod,DBDEFAULT]]"
Set VARIABLE TIMEPERIODS = ListAppend(TIMEPERIODS, DIFF")
Set VARIABLE TIMEPERIODS = ListAppend(TIMEPERIODS, "
[[SYSTEM,SGPBudgetPeriod,DBDEFAULT]]")
```

To capture additional user input via a form, it is required that the button clicked be copied to the FORM_ ButtonClicked variable to ensure expected app execution. Specifically, app execution continues if FORM_ButtonClicked has a value of “OK”.

Example: Capture additional user input via a form

```
Show FORM USING Custom.lvfrm
Set VARIABLE FORM_ButtonClicked = $LVS_BUTTONCLICKED$
```

Allowing currency re-selection in a data view input app

There may be cases where the initial selection in the app is based on the functional currency of the selected entity, but you want the user to be able to change the currency.

To accomplish this, follow these steps:

Note: Substitute CURRENCY for the name of your currency dimension, or use an attribute token or $CORE.DimensionInfo.KeyDimensionList[4]$ for CURRENCY for flexibility.

1. In the pre-selection add the following to support currency selection:

```
//Configure currency selections for re-select in view
Create GLOBALVARIABLE CURRENCYDefault AS STRING
Create GLOBALVARIABLE CURRENCYOptions AS STRING
Create GLOBALVARIABLE CURRENCYSymbols AS RANGE
Create GLOBALVARIABLE oldSelectDimensionList[] AS STRING //used to
restore symbol selection to original options
```

```
Set VARIABLE CURRENCYOptions = "11000"
Set VARIABLE CURRENCYSymbols = "SOURCEC###"

Create VARIABLE isCurrencySelection AS NUM //used to track in post-
selection if the is changing currency only
```

2. In the Post-Selection add logic to set the native currency on entity selection:

```
If Not(VariableExists("CURRENCY"))
    Create VARIABLE CURRENCY AS STRING
END If

If $isCurrencySelection$ //reset selection options to allow regular
symbol selections
    Set VARIABLE FORM_SelectDimensionList = ListAppend(CORE_EmptyList,
oldSelectDimensionList)
    Set VARIABLE isCurrencySelection = 0
Else
    Set VARIABLE CURRENCY = "[[SYMBOL,ZGPNativeCurrency,$ENTITIES$]]"
//apply currency of selected entity
END If
```

3. Add a View Action to allow the user to change currency (this will be a new button on the toolbar)

```
Set VARIABLE oldSelectDimensionList = ListAppend(CORE_EmptyList, FORM_
SelectDimensionList)
Set VARIABLE isCurrencySelection = 1

Set VARIABLE FORM_SelectDimensionList = ListAppend(CORE_EmptyList,
"CURRENCY")

Run PROCEDURE "VIEW\OnReselect.lvpro"
```

Specifying settings

You can use Longview Designer to specify settings for a validation report app.

Field	Notes
Enter Dimensions to Display	Select the dimensions that you would like to display in the report.
Enter Validation Threshold	Enter a numeric thereshold value. Any validation differences that do not meet the threshold (either + or -) will not be displayed in the validation report. For example, if a threshold of 1000 is entered, validation differences less than 1000 will not display.
Show Difference as Absolute Value	Select Yes to display the validation differences as absolute values.
Validation Data Area	Set the Data Area that you would like the validation report to run against. **Note:** ■ The ACCOUNTS dimension should be set to the appropriate validation reporting symbols in your system (e.g.: ACCOUNTS_ValRep). ■ For other dimensions make sure the symbols specified are consistent with the underlying validation rule. For example, if a validation rule is specified on year-to-date time periods validation reuslts will only appear on those time periods and not on any related period activity time periods.

Creating Events

You can use event templates to build event-based calculations. When creating an event through an event template, all procedures, models, data areas and data table definitions will be saved within a kar file on the Longview application server.

- The name of the kar file created will have the same name as the name provided for the event itself.
- The kar file will be placed in the Longview application server's 'Events' folder. For example: Longview\DataServers\<LID>\applications\Events.
- The kar file will contain the calling procedure named <EventName>\Event.lvpro; where <EventName> is the name provided for the event itself.
- An event rule would need to be created to run the event. The event rule syntax would need to be set to run <EventName>\Event.lvpro.

Example:

```
KLX(EVNT_6000100,Actual_Open###,ENTITIES_ALL###,DATAVIEWS_Default,SCENARIOS_
Actual###,CURRENCIES_Default)=RUNPROC
("applications\Events\SampleEvent\Event.lvpro")
```

Specifying the manual trigger override

Use Manual Trigger Override to specify any commands to run if the event is triggered manually. The manual trigger override is typically used to set any E_<DIMNAME> variables where the symbol to be calculated in the event is different than the symbol set up in the manual trigger event rule.

For example, for a Longview Close system, normally a manual trigger event rule is set to use CURRENCIES_Default as the currencies symbol. However, the calculation itself is normally performed on all currencies. The default code block in the manual trigger override sets the E_CURRENCIES variable to all currencies.

For a Longview Tax system, normally the Load Parameters procedure is to retrieve all the parameters required for a manually triggered event. The E_ENTITIES variable is also normally filtered to remove the DIM2SET entity symbol from the list of entities.

Specifying the additional KAR files

Additional KAR files can be included in your event by adding them within the additional KAR files section of the template. Once the KAR files are included, any documents within the KAR file can be executed, such as procedures or models.

Normally, additional KAR files are used to create common logic that can be shared among multiple events.

To specify additional KAR files:

1. Right-click on Additional KAR files and click Add KAR file.
2. For the KAR file name, enter the name of the kar file. For example, Sample.kar

   Note: The default path for the KAR file is under the Longview Data Server applications directory. If a different path is required, the path must be included in the KAR file name. For example, Libraries\Sample.kar.

3. The Root Path is the path that is virtually prepended to all files in the .kar. This is optional and can be left blank.

Specifying the main procedure

The Main Procedure is the controlling procedure that is run when the event is triggered. Use the main procedure to configure your event. You can use the template document “Template_Event_Main” to provide a framework for your main event procedure.

If you are creating a persistent event, you may want to make use of the LVS_EVENTINITIALIZE, LVS_EVENTSHUTDOWN and LVS_PERSISTENT system variables to improve the efficiency and speed of your events. You can use the template document “Template_Event_Persistent” to provide a framework for using these system variables in your code.

For more information, see Working with System Variables for Persistent Event Rules.

Specify the data area definition

Use Data Area Definition to specify the data area to be queried for the export. Open the data area definition to specify its properties.

Note: The name of the data area definition is Main.lvdsp and the name of the data area created using it is daMain.

1. In the Options table, complete the following fields:

Field	Notes
Data type	Specify the data to be queried using one of the following options: ■ ADJUSTED — Indicates that data queries include adjustments made via journal entries ■ UNADJUSTED — Indicates that data queries exclude adjustments made via journal entries Default is ADJUSTED.
Download type	Specify the type of data to be downloaded using one of the following options (download option specific in brackets): ■ All data — Indicates that all data within the data area definition is downloaded (STANDARDALL). ■ Exclude parent values — Indicates that parent data is excluded in the download (STANDARDLEAFONLY) ■ Exclude parent and calculated values — Indicates that only leaf data that was submitted to the database via import, input or event calculation is included in the download (LEAFDATA). Default is All data.

2. In the name field, enter the name of your Data Area Definition
3. In the Data Area table, complete the following fields for each dimension:

Note: The Spec and Level fields are not applicable if a symbol contains a variable.

Field	Notes
Symbols	Type a symbol name. Alternatively, you can use the symbol selector. To add more than one symbol for a dimension, select a row containing the dimension for which you want to add another symbol and click Symbol. To move a symbol up or down within a dimension, click Move Up or Move Down. **Note:** To use the symbol selected by the user or defined by an attribute, enter the name of the dimension as a variable, for example $ENTITIES$

Field	Notes
Spec	Indicate the symbol specification using one of the following values: ■ All: Includes all symbols within the selected symbol's hierarchy. ■ Leaf: Includes only leaf symbols within the selected symbol's hierarchy. ■ Parent: Includes only parent symbols within the selected symbol's hierarchy. ■ Root and Parent: Includes only root and parent symbols within the selected symbol's hierarchy. **Note:** To use the symbol selected by the user or defined by an attribute, enter the name of the dimension as a variable, for example $ENTITIES$
Level	Specify the number of levels down from the symbol to be included in the data area.

4. Use Additional Configuration to specify any other Data Area Definition features you wish to use. Additional Configuration is a code editor that allows you to enter data area definition functions to be included in the data area.

   For more information on the code editor, see Using the Code Editor.

   Typical use of Additional Configuration in a data area definition is to

   - Add an attribute filter using the AttributeFilter function
   - Add a schedule to the using the Schedule function
   - Add a temporary symbol using the TempSym function

Specifying procedures

Use Procedures to create any additional procedures that are required as part of the app, event, or configuration library.

To create a procedure:

1. Right-click on **Procedures** and select **New Document**.
2. For the Name, enter the name of your Procedure.
3. In the code editor, enter the commands to run when the procedure is executed. For more information on procedures, see Developing Longview Procedures.

Specifying models

Use Models to create models that are executed as part of the app, event, or configuration library.

To create a model:

1. Right-click on **Models** and select **New Document**.
2. For the Name, enter the name of your Model.

3. In the code editor, enter the code required for the model calculation. For more information on models, see Developing Longview Models.

Specifying table definitions

To create a table definition:

1. Right-click on Table Definition and select New Document.
2. For the Name, enter the name of your Table Definition.
3. In the columns section the table and columns are defined.
    - For the table name, select the table to use. The list includes all app tables defined in the system.
    - If you pick a table based on a view it will display an information icon indicating that the table will be read-only.
    - After selecting a table the columns in the table will be displayed. All columns will be selected by default. Also the columns that define the primary key will be indicated with a key icon. If some or all of the primary key columns are not included, the table will be read only.
    - You may add or remove temporary columns using the Add Column and Remove Column buttons.

      Note: Temporary columns are added to the table for use with an app, but are not submitted to the database.

    - For each column complete the following fields:

Field	Notes
Include	Check each column to be included in the query. Use the check box in the header to select all columns.
Column	For persistent table columns, displays the name of the column. For temporary columns, enter the name of the column.
Nullable	Indicates whether the column can be saved with no value specified. If any non-nullable columns are excluded the table will be read-only.
Type	For persistent table columns, displays the data type of the column. For temporary columns, select the data type for the column. If you select Symbol[Dimension], you will be prompted to select the dimension that applies. If you select Symbol[DimensionColumnName], you will be prompted to select a column (of type Dimension).

Field	Notes
Values (String)	For String columns allows you to optionally define a list of values the user can select from. Enter a | delimited list of values. Optionally enter a | delimited list of display values by placing a semi-colon after the list of values. **Note:** You can also use variables to define the list of values and display values.
Values (Symbol)	For Symbol columns allows you to limit the symbols available. Enter a | delimited list of symbol specifications. You can also use a form to select the symbol specifications by clicking the search icon. **Note:** You can also use a variable to define the list of symbol specifications.
Values (User)	For User columns, allows you to limit the users available for selection. Enter (or select) a | delimited list of user and group names. **Note:** You can also use a variable to define the list of users and groups.
Values (User List)	For UserList columns, allows you to limit the users available for selection. Enter (or select) a | delimited list of user and group names. **Note:** You can also use a variable to define the list of users and groups.
Allow Override	For String columns with Values defined, indicates whether the user can enter a value that is not in the list of values.
Min / Max (Date)	For date columns, allows you to set a valid range of selectable dates. **Note:** The interface only supports explicit dates. To specify the valid date range using variables use the Additional Configuration section to define the date range using the SetColumn function.
Default (Boolean)	For Boolean columns, allows you to indicate if the default value for the column is checked.
Default (Dimension)	For Dimension columns, allows you to set he default dimension to be selected when a new row is added. Default is no default selection.

Creating Event Trigger Apps

You can use Longview Designer to create event trigger apps.

Specifying symbol selections no pre-post

Specifying user selections

Use user selections to specify dimensions for which the user will be prompted to choose symbol(s). For each dimension that the user will choose symbols, right-click on the User Selections folder and choose Add Dimension. The dimension settings page will appear.

Variable	Notes
Dimension	Select the dimension the user will be prompted to select symbol(s) for.
Available selections	Select a symbol to limit the possible symbols available to the user. If left blank the user will be able to select from all symbols available in the dimension. The available symbols will always be limited by user access.
Default selection	Select a symbol to specify as the default selection. This can either be a database symbol, an attribute symbol, or a floating time period. **Note:** Attribute symbols and floating time periods will only be available if they are defined in your system. For more information, see “Specifying attributes in Longview Application Administrator” in the *Longview Analysis and Reporting Guide*.
Selection required	Check this to force the user to select a symbol to continue.
Allow leaf selection	Check this to allow the user to select a leaf symbol.
Allow parent selection	Check this to allow the user to select a parent symbol.
Allow read-only selection	Check this to allow the user to select a read-only symbol.
Allow multiple selections	Check this to allow the user to select more than one symbol.
Attribute filter	Enter an attribute filter expression to further limit the possible symbol selections. For more information see SymbolSelector in the *Longview Developer’s Guide*.

Specifying attribute-based selections

Use Attribute Based Selections to specify selections for dimensions that are deter- mined by an attribute value. For each dimension to be determined via an attribute, right-click on Attribute Based Selections and select Add Dimension.

Variable	Notes
Dimension	Select the dimension to select symbol(s) using an attribute value.
Attribute class	Select the class of attribute to use.
Attribute name	Select the attribute.
Source dimension (SYMBOL class)	If the attribute class is SYMBOL, select the source dimension. **Note:** The source dimension must be a dimension specified as a user selection or attribute-based selection.

Note: The variable created to hold the attribute-based symbol will have the same name as the dimension. The variable created will be a STRING type.

Example, to set the currency selection to be the functional currency of the selected entity:

1. Select the CURRENCIES dimension.
2. Set the attribute class to SYMBOL.
3. Enter “ZGPNativeCurrency” for the attribute name.
4. Select ENTITIES for the source dimension.

Example, to set the time period selection to be the current period:

1. Select the TIMEPERIODS dimension.
2. Set the attribute class to SYSTEM.
3. Enter “SGPCurrentPeriod” for the attribute name.

Specify the trigger definitions

Use Trigger Definitions to specify the events to be triggered and the specific symbols to trigger the event for. For each trigger definition to be created, right-click on Trigger Definitions and select New Document.

1. For the Name, enter the name of your Trigger Area

   Note: The name must 29 or fewer characters and only contain alphanumeric characters and underscore “_”.

2. In the Event table, complete the following fields:

Field	Notes
Event to trigger	Select the event to trigger. This selection corresponds to the account dimension symbol used to trigger the manual trigger event rule.

3. In the Trigger Area table, complete the following fields for each dimension:

> **Note:** The Spec and Level fields are not applicable if a symbol contains a variable.

Field	Notes
Symbols	Type a symbol name. Alternatively, you can use the symbol selector. To add more than one symbol for a dimension, select a row containing the dimension for which you want to add another symbol and click Symbol. To move a symbol up or down within a dimension, click Move Up or Move Down. **Note:** To use the symbol selected by the user or defined by an attribute, enter the name of the dimension as a variable, for example $ENTITIES$
Spec	Not used
Level	Not used

Calculated Journal Entries

You can use Longview Designer to create, duplicate, modify, and delete calculated journal entry (CJE) events.

Calculated journal entry event template overview

You can use Calculated Journal Entry event templates to build event-based calculated journal entries. When creating a Calculated Journal Entry event through a Calculated Journal Entry event template, all procedures, models, data areas and data table definitions will be saved within a kar file on the Longview application server.

- The name of the kar file created will have the same name as the name provided for the event itself. See Specifying properties.
- The kar file will be placed in the Longview application server's 'Events' folder. For example: Longview\DataServers\<LID>\applications\Events.
- The kar file will contain the calling procedure named <EventName>\Event.lvpro; where <EventName> is the name provided for the event itself. See Specifying properties.
- An event rule would need to be created to run the event. The event rule syntax would need to be set to run <EventName>\Event.lvpro.

Example:

```
KLX(EVNT_6000100,Actual_Open###,ENTITIES_ALL###,DATAVIEWS_Default,SCENARIOS_
Actual###,CURRENCIES_Default)=RUNPROC
("applications\Events\SampleEvent\Event.lvpro")
```

Calculated journal entry event template - process flow

When creating a calculated journal entry event, there are a few different areas to configure. The following describes the process flow of how the pieces work together and the order of operations of their execution. These areas will be discussed in more detail in Creating calculated journal entry events

Event entry point

The entry point for the calculated journal entry event is the Event procedure (<EventName>\Event.lvpro). This procedure is included in every calculated journal entry event and is not directly editable in Designer. The event procedure performs the following functions:

1. Initializes Calculated Journal entry variables. See CJE global variables.
2. Determines if the calculated journal entry event is manually triggered or data trigger. If it is manually triggered, it will run the Manual Trigger Override commands. For more information, see Specifying the manual trigger override.
3. Proceeds to run the Main Procedure.

Manual Trigger Override

Commands specified to run if the event is triggered manually.

For more information, see Specifying the manual trigger override.

Main Procedure

The main procedure is executed once the manual trigger override commands are run. Default code is provided and described in detail in Specifying the main procedure.

The main procedure controls the flow of the calculated journal entry event. Within the main procedure, and at the point where you want to process the calculated journal entry, procedure “CJE\ProcessCJE.lvpro” should be called.

ProcessCJE Procedure

The CJE\ProcessCJE.lvpro performs two steps:

1. Runs the Set Values procedure. This is procedure where the actual values for the calculated journal entry are set. For more information, see Specifying the set values procedure.
2. Determines whether the calculated journal entry requires posting and unposts, creates and posts the calculated journal entry as required.

Set Values Procedure

The Set Values Procedure is the procedure where the values for the calculated journal entry detail variables are set. Within the Set Values procedure, variable CJE_IsPostRequired should be set to 1 when posting is required and set to 0 when posting is not required.

For more information, see Specifying the set values procedure.

Calculated JE Detail and Header

The settings (Specifying calculated journal entry settings) and Calculated JE Detail (Specifying calculated journal entry detail) are used to build the file used for posting. The Calculated JE Detail should contain the variables used for the values in the Set Values procedure.

Creating Calculated Journal Entry Events

You can use Longview Designer to create Calculated Journal Entry events.

Specifying calculated journal entry settings

You can use Longview Designer to specify the main settings for a Calculated Journal Entry event.

Field	Notes
Application ID Prefix	Type the name you want to use as the prefix for the calculated journal entry application ID. The application ID is the combination of the specified application ID prefix, the entity name (optional), two digits for the creation year indicator and two digits for the creation month indicator. For example, NCI_727_1610, where NCI would be the Application ID Prefix. The complete application ID name can have a maximum of 31 characters.
JE Description	Type a description for the Calculated Journal Entry. The description can have a maximum of 100 characters, including spaces.
JE Notes	Type any detailed notes for the Calculated Journal Entry. This is an optional parameter. The notes field can have a maximum of 1280 characters, including spaces.
JE Subcategory	Type a valid journal entry sub-category. If no sub-category is chosen, Adjustment is used by default.
Create a calculated journal entry by entity	Select this option if a journal entry by entity is needed. **Note:** If this option is selected, the application ID will include the entity name.
Is this a financial JE?	Specify if the journal entry is financial: ■ Yes — the journal entry must be in balance. ■ No — the journal entry is not required to be in balance. The default setting is Yes.
Is this a Shared JE?	Specify if the journal entry is shared: ■ Yes — the journal entry will be shared. ■ No — the journal entry is not be shared. The default setting is Yes.

Specifying the manual trigger override

Use Manual Trigger Override to specify any commands to run if the event is triggered manually.

The manual trigger override is typically used to set any E_<DIMNAME> variables where the symbol to be calculated in the event is different than the symbol set up in the manual trigger event rule.

For example, normally a manual trigger event rule is set to use CURRENCIES_Default as the currencies symbol. However, the calculation itself is normally performed on all currencies. The default code block in the manual trigger override sets the E_CURRENCIES variable to all currencies.

Specifying the main procedure

The Main Procedure is the controlling procedure that is run when the event is triggered. A standard code template is provided and can be modified. The standard code provided performs the following:

1. Creates a data area where the calculations will be performed

   Note: The template code creates the data area using Main.lvdsp. This data area spec will need to be created. See Specifying the data area definition for more information

2. Processes each time period and determines:

   - If it is the current period, the calculated journal entry type would be CURRENTPERIOD
   - If it is not the current period, the calculated journal entry type would be PPA (Prior Period Adjustment)

3. Processes each entity individually and adds the entity name to the APP ID if 'Create a calculated JE by entity' is selected. For more information, see Specifying Calculated Journal Entry Settings.

   Note: The application ID is the combination of the specified application ID prefix, the entity name (optional), two digits for the creation year indicator and two digits for the creation month indicator. The complete application ID name can have a maximum of 31 characters. If the entity name is too long and would cause the application ID to be over 31 characters, the entity name will be truncated in the application ID.

4. Runs the "CJE\ProcessCJE.lvpro" procedure which handles processing, creating and posting the calculated journal entry. For more information on the flow of the ProcessCJE procedure, see Calculated Journal Entry Event Template - Process Flow.

   Note: If you are creating a persistent event, you may want to make use of the LVS_EVENTINITIALIZE, LVS_EVENTSHUTDOWN and LVS_PERSISTENT system variables to improve the efficiency and speed of your events. You can use the template document "Template_Event_Persistent" to provide a framework for using these system variables in your code. For more information, see Working with System Variables for Persistent Event Rules.

Specifying the set values procedure

The Set Values Procedure is the procedure where the values for the calculated journal entry are determined. The values should be stored in variables which would be used in the calculated journal entry detail.

For more information, see Specifying Calculated Journal Entry Detail.

Within the Set Values procedure, variable CJE_IsPostRequired should be set to 1 when posting is required, and set to 0 when posting is not required.

If a journal entry is required to be unposted, deleted or created and posted within the Set Values procedure, the following procedures can be used:

- CJE\UnPost.lvpro
- CJE\Delete.lvpro
- CJE\CreateAndPost.lvpro

Prior to running these procedures, the variable CJE_Status should be checked to make sure the journal entry is eligible to be unposted, deleted, or created and posted. The following CJE_Status values are valid for each action:

Action	Valid CJE_Status value
Unpost	5 = Review
Delete	1 = No Status 3 = Validated
Create and Post	0 = Does Not Exist 1 = No Status 3 = Validated

Each procedure will reset the CJE_Status variable upon completion.

You can use the template document “Template_CJE_Process_Line” to provide a framework for your set values procedure.

For more information on Template Documents, see "Template documents".

Example:

```
Set VARIABLE credDebAmt1 = Round(Value("daMain", "7500", "$CJE_PostPeriod$",
"737", "GL", "SCENARIOS_A001", "USDT"), 2)

Set VARIABLE credDebAmt2 = Round(Value("daMain", "7750", "$CJE_PostPeriod$",
"878", "GL", "SCENARIOS_A001", "USDT"), 2)

Set VARIABLE credDebTotal = $credDebAmt1$ + $credDebAmt2$

Set VARIABLE CJE_IsPostRequired = Round($credDebTotal$, 2) != 0
```

Specifying calculated journal entry detail

Use Calculated JE Details to add detail lines that the calculated journal entry event will use to post. The header of the calculated journal entry is handled internally based on the settings set in Specifying calculated journal entry settings.

You can use the Detail template for syntax reference by typing in Template in the Calculated JE Detail.

For more information on Calculated journal entry documents, see “Creating CalculatedJESpec documents” in the *Longview Developer’s Guide*.

Example:

```
Detail DEBIT, $credDebAmt1$, 7500, 737A, CONS_IC, SCENARIOS_A001, USD
Detail CREDIT, $credDebAmt1$, FXIC, 737A, CONS_IC, SCENARIOS_A001, USD
Detail DEBIT, $credDebAmt2$, 7750, 878A, CONS_IC, SCENARIOS_A001, USD
Detail CREDIT, $credDebAmt2$, FXIC, 878A, CONS_IC, SCENARIOS_A001, USD
```

Specify the data area definition

Use Data Area Definition to specify the data area to be queried for the export. Open the data area definition to specify its properties.

Note: The name of the data area definition is Main.lvdsp and the name of the data area created using it is daMain.

1. For the Name, enter the name of your Trigger Area

   Note: The name must 29 or fewer characters and only contain alphanumeric characters and underscore “_”.

2. In the Options table, complete the following fields:

Field	Notes
Data type	Specify the data to be queried using one of the following options: ■ ADJUSTED — Indicates that data queries include adjustments made via journal entries ■ UNADJUSTED — Indicates that data queries exclude adjustments made via journal entries Default is ADJUSTED.
Download type	Specify the type of data to be downloaded using one of the following options (download option specific in brackets): ■ Exclude parent values — Indicates that parent data is excluded in the download (STANDARDLEAFONLY) ■ Exclude parent and calculated values — Indicates that only leaf data that was submitted to the database via import, input or event calculation is included in the download (LEAFDATA). Default is All data.

3. In the Trigger Area table, complete the following fields for each dimension:

   Note: The Spec and Level fields are not applicable if a symbol contains a variable.

Field	Notes
Symbols	Type a symbol name. Alternatively, you can use the symbol selector. To add more than one symbol for a dimension, select a row containing the dimension for which you want to add another symbol and click Symbol. To move a symbol up or down within a dimension, click Move Up or Move Down. **Note:** To use the symbol selected by the user or defined by an attribute, enter the name of the dimension as a variable. For example, $ENTITIES$.
Spec	Indicate the symbol specification using one of the following values: ■ All: Includes all symbols within the selected symbol's hierarchy. ■ Leaf: Includes only leaf symbols within the selected symbol's hierarchy. ■ Parent: Includes only parent symbols within the selected symbol's hierarchy. ■ Root and Parent: Includes only root and parent symbols within the selected symbol's hierarchy. **Note:** To use the symbol selected by the user or defined by an attribute, enter the name of the dimension as a variable. For example, $ENTITIES$.
Level	Specify the number of levels down from the symbol to be included in the data area.

4. Use Additional Configuration to specify any other Data Area Definition features you wish to use. Additional Configuration is a code editor that allows you to enter data area definition functions to be included in the data area.

   For more information on the code editor, see Using the Code Editor.

   Typical use of Additional Configuration in a data area definition is to

   - Add an attribute filter using the AttributeFilter function
   - Add a schedule to the using the Schedule function
   - Add a temporary symbol using the TempSym function

Specifying procedures

Use Procedures to create any additional procedures that are required as part of the app, event, or configuration library.

To create a procedure:

1. Right-click on **Procedures** and select **New Document**.
2. For the Name, enter the name of your Procedure.
3. In the code editor, enter the commands to run when the procedure is executed. For more information on procedures, see Developing Longview Procedures.

Specifying models

Use Models to create models that are executed as part of the app, event, or configuration library.

To create a model:

1. Right-click on **Models** and select **New Document**.
2. For the Name, enter the name of your Model.
3. In the code editor, enter the code required for the model calculation. For more information on models, see Developing Longview Models.

Specifying table definitions

To create a table definition:

1. Right-click on Table Definition and select New Document.
2. For the Name, enter the name of your Table Definition.
3. In the columns section the table and columns are defined.
    - For the table name, select the table to use. The list includes all app tables defined in the system.
    - If you pick a table based on a view it will display an information icon indicating that the table will be read-only.
    - After selecting a table the columns in the table will be displayed. All columns will be selected by default. Also the columns that define the primary key will be indicated with a key icon. If some or all of the primary key columns are not included, the table will be read only.
    - You may add or remove temporary columns using the Add Column and Remove Column buttons.

      Note: Temporary columns are added to the table for use with an app, but are not submitted to the database.

- For each column complete the following fields:

Field	Notes
Include	Check each column to be included in the query. Use the check box in the header to select all columns.
Column	For persistent table columns, displays the name of the column. For temporary columns, enter the name of the column.
Nullable	Indicates whether the column can be saved with no value specified. If any non-nullable columns are excluded the table will be read-only.
Type	For persistent table columns, displays the data type of the column. For temporary columns, select the data type for the column. If you select Symbol[Dimension], you will be prompted to select the dimension that applies. If you select Symbol[DimensionColumnName], you will be prompted to select a column (of type Dimension).
Values (String)	For String columns allows you to optionally define a list of values the user can select from. Enter a | delimited list of values. Optionally enter a | delimited list of display values by placing a semi-colon after the list of values. **Note:** You can also use variables to define the list of values and display values.
Values (Symbol)	For Symbol columns allows you to limit the symbols available. Enter a | delimited list of symbol specifications. You can also use a form to select the symbol specifications by clicking the search icon. **Note:** You can also use a variable to define the list of symbol specifications.
Values (User)	For User columns, allows you to limit the users available for selection. Enter (or select) a | delimited list of user and group names. **Note:** You can also use a variable to define the list of users and groups.

Field	Notes
Values (User List)	For UserList columns, allows you to limit the users available for selection. Enter (or select) a | delimited list of user and group names. **Note:** You can also use a variable to define the list of users and groups.
Allow Override	For String columns with Values defined, indicates whether the user can enter a value that is not in the list of values.
Min / Max (Date)	For date columns, allows you to set a valid range of selectable dates. **Note:** The interface only supports explicit dates. To specify the valid date range using variables use the Additional Configuration section to define the date range using the SetColumn function.
Default (Boolean)	For Boolean columns, allows you to indicate if the default value for the column is checked.
Default (Dimension)	For Dimension columns, allows you to set he default dimension to be selected when a new row is added. Default is no default selection.

Creating Free Form Apps

You can use Longview Designer to create free form apps.

Specifying properties

You can use Longview Designer to specify properties for a new free form app.

For more information, see Specifying properties.

Specifying troubleshooting settings

You can use troubleshooting settings and tools to track down issues in a free form app.

For more information see Setting troubleshooting options.

Specifying the execute procedure

You can use Longview Designer to specify the execute procedure for your free form app. The Execute procedure uses commands to control the flow of the free form app.

Specifying Calculated Journal Entry Specs

A Calculated Journal Entry Spec (.lvcje) is an ASCII file that specifies information about a calculated journal entry such as the journal entry type and description, as well as a line of detail for each adjustment that the calculated journal entry contains.

For more information, see “Developing Longview CalculatedJESpecs” in the *Longview Developer’s Guide*.

For more information on using the code editor, see Using the code editor.

Specifying Data Area Definitions

Use Data Area Definitions to create a dataspec document (.lvdsp) used to define an area in the database. For each data area definition to be created, right-click on Data Area Definitions and select New Document.

Note: If you would like to use a code editor instead of a UI to create the Data Area Definition, see Specifying Data Area Definitions – Script.

1. For the Name, enter the name of your Data Area
2. In the Options table, complete the following field

Field	Description
Data Type	Specify the data to be queried using one of the following options: ■ ADJUSTED — Indicates that data queries include adjustments made via journal entries ■ UNADJUSTED — Indicates that data queries exclude adjustments made via journal entries Default is ADJUSTED

3. In the Data Area table, complete the following fields for each dimension:

Note: The Spec and Level fields are not applicable if a symbol contains a variable.

Field	Description
Symbols	Type a symbol name. Alternatively, you can use the symbol selector. To add more than one symbol for a dimension, select a row containing the dimension for which you want to add another symbol and click Symbol. To move a symbol up or down within a dimension, click Move Up or Move Down.
Spec	Indicate the symbol specification using one of the following values: ■ All — Includes all symbols within the selected symbol's hierarchy. ■ Leaf — Includes only leaf symbols within the selected symbol's hierarchy. ■ Parent — Includes only parent symbols within the selected symbol's hierarchy. ■ Root and Parent — Includes only root and parent symbols within the selected symbol's hierarchy.
Level	Specify the number of levels down from the symbol to be included in the data area.

4. Use Additional Configuration to specify any other Data Area Definition features you wish to use. Additional Configuration is a code editor that allows you to enter data area definition functions to be

included in the data area.

For more information on the code editor, see Using the code editor.

Typical use of Additional Configuration in a data area definition would be to

- Add an attribute filter using the AttributeFilter function
- Add a schedule to the using the Schedule function
- Add a temporary symbol using the TempSym function

Specifying Data Area Definitions - Script

Data Area Definitions - Script allows you to create a DataSpec document (.lvdsp) using a code editor. This defines an area in the database.

For more information, see “Developing Longview DataSpecs” in the *Longview Developer’s Guide*.

For more information on using the code editor, see Using the code editor.

Note: If you would like to use a UI instead of a code editor to create the Data Area Definition, see Specifying Data Area Definitions.

Specifying Data Area Views

Use Data Area Views to create a data view document (.lvdvw) used to specify how the data areas will be displayed to the user. For each data area view to be created, right-click on Data Area Views and select New Document.

Note: If you would like to use a code editor instead of a UI to create the Data Area Views, see Specifying Data Area Views - Script.

Name

Specify the name of the data area view

Setting view orientation

Use orientation to determine how each dimension is presented to the user. At least one dimension must be in the rows and columns orientation. Multiple dimensions in rows or columns are nested outer to inner, based on their left to right position.

Use the following table to determine the way dimensions display in the view.

Dimension orientation	View appearance
Columns or Rows	The left-most dimension listed in the orientation section is the outermost dimension in the view.

Dimension orientation	View appearance
Slice	Each dimension in the slice dimension is presented to the user as a symbol selector. Dimensions are presented in the view in the order that they appear in the Slice orientation.
Fixed	The view opens displaying fixed symbols in dimension order on the Fixed Symbols tab. Dimensions appear on the Fixed Symbols tab in their natural order regardless of the order listed in the Fixed orientation.

To modify dimension orientation:

1. In the Orientation section for the dimension, you want to reorient, select the dimension.
2. Drag and drop the selected dimension into the desired position.

Specifying symbols

Use symbol selection to specify the symbols that are displayed in the view.

In the Symbol Selection table, complete the following fields for each dimension: The Spec and Level fields are not applicable if a symbol contains a variable.

Field	Description
Symbols	Type a symbol name. Alternatively, you can use the symbol selector. To add more than one symbol for a dimension, select a row containing the dimension for which you want to add another symbol and click Symbol. To move a symbol up or down within a dimension, click Move Up or Move Down.
Spec	Indicate the symbol specification using one of the following values: ■ Root and Parent — Includes only root and parent symbols within the selected symbol's hierarchy. ■ All — Includes all symbols within the selected symbol's hierarchy. ■ Leaf — Includes only leaf symbols within the selected symbol's hierarchy. ■ Parent — Includes only parent symbols within the selected symbol's hierarchy. ■ Root and Parent — Includes only root and parent symbols within the selected symbol's hierarchy.
Level	Specify the number of levels down from the symbol to be included in the data area.
Levels to Expand	For dimensions in the rows and column orientation only, you can specify the number of levels to expand the hierarchy of each symbol included.

Specifying dimension display settings

Use dimension display settings to specify how symbols are displayed in the view. In the Dimension Display Settings table, complete the following fields for each dimension:

Note: Show Name and Show Description only affect dimensions in the Rows or Columns orientation.

Field	Description
Show Name	Check this to display the name of the symbol in the view. If both Show Name and Show Description are selected the symbol name and description will be displayed separated by a hyphen for symbols in the dimension.
Show Description	Check this to display the description of the symbol in the view. If both Show Name and Show Description are selected the symbol name and description will be displayed separated by a hyphen for symbols in the dimension.
Fit to Title	Check this to make the view auto-size to fit symbol titles. For Rows this affects the width of the title. For Columns this can only be selected for the inner dimension and affects the width of the column.
Column Width	If Fit to Title is not selected, enter the width in pixels.
Wrap	If Fit to Title is not selected, check this to wrap titles onto new lines if the text does not fit in the specified width.

Specifying number formatting

Use Number Formatting to specify how numbers are displayed in the view.

In the Number Formatting table complete the following fields:

Field	Description
Default decimals	Enter the number of decimals to display for each numeric value. This should be from 0 to 9. This can be overridden for a specific symbol using the SymbolDecimals function in the Additional Configuration section.
Use parentheses for negative numbers	Select Yes to display negative numbers with parentheses. Select No to display negative numbers with a negative sign.
Use thousands separator for numbers	Select Yes to display a locale-specific thousands separator based on the operating system setting. Select No to not display a thousand separator.

Specifying options

Use Options to specify additional view settings.

In the Options table complete the following fields:

Field	Description
Allow dynamic rollup of input	Select Yes to have parent totals automatically calculated as the user enters numeric values.

Customizing the data view

Use Additional Configuration to specify any other view features you wish to use in the data view. Additional Configuration is a code editor that allows you to enter view functions to be included in the configuration of the view.

For more information on the code editor, see Using the code editor.

Typical use of Additional Configuration in this app would be to

- Add dynamic calculations to the view using the DynamicModel function
- Add protected areas to the view using the Protect function
- Add conditional style to the view using the ConditionalStyle function
- Restrict input to numeric values or text values using the NumericInputOnly and/or TextInputOnly functions
- Hiding specific symbols in the view using the Hide function
- Overriding the decimals displayed for certain symbols using the SymbolDecimals function
- Adding separators between symbols using the InsertSeparator function

Specifying Data Area Views - Script

Data Area Views - Script allows you to create a Data View document (.lvdsp) using a code editor. A Data View document defines the way in which the data area is rendered as a Data Grid.

For more information, see “Designing Longview Data Grids” in the *Longview Developer’s Guide*.

For more information on using the code editor, see Using the code editor.

Note: If you would like to use a UI instead of a code editor to create the Data Area Views, see Specifying Data Area Views.

Specifying Data Area Export Specs

Data area export specs (.lvexp) can be run from a procedure to export data to a file. The target file may contain a single column of values or multiple columns of values.

Note: If you would like to use a code editor instead of a UI to create the Data Area Export Spec, see Specifying Export Specs – Script.

To create a data area export spec:

1. Right-click on one of the Data Area Export Specs folders and select New Document.
2. For the Name, enter the name of your Export Spec.
3. Fill in the data target
4. In the File Options table, complete the following fields:

Field	Description
Target File	Specify the path and name of the data target.
Field Delimiter	Specify whether the character used to separate fields in the data source is the brace ({), comma (,), semicolon (;), pipe (|), or regional delimiter for comma separated values (Regional). Alternatively, you can type a field delimiter of your choice. The default is the regional delimiter for comma separated values.

5. For a multiple value, file-based export: in the Values Dimension table complete the following field:

Field	Description
Dimension corresponding to values	Specify the dimension for which there are multiple value fields.

6. The Dimension Definitions section is used to specify how each dimension is processed during export. For Dimension Definitions, complete the following:
   - Schedule: This field displays only if your system is configured to use schedules and you have access to schedules.
     a. The default of None indicates the export spec is for base data only.
     b. Specify the schedule to use for the import spec, and the schedule dimensions appear at the bottom of the list of Dimensions.
   - For each dimension, complete the following fields:

Field	Description
Dimension	The name of the dimension.

Field	Description
Type	Specify the type of relationship between dimension symbols and field values in the data source using one of the following values: • Map — The dimension symbols are mapped to values as specified in the indicated map • Match — The dimension symbols are written out with their names. • Unique — Values for the symbol specified are exported, but the symbol name is not written in the target file. **Note:** For a multiple values export spec, the dimension you indicated has Values for the Type field and cannot be changed.
Field Name	This field is available only when Type is Map or Match. Optionally, enter the name of the field. If the field name is not specified it will be set to Field with the Field Position appended. The field name is used to specify filters.
Field Position	This field is available only when Type is Map or Match. Specify the field position to assign the dimension to using a positive integer. For example, for the Accounts dimension, type 3 if the account will be the third column of the data target.
Symbol	This field is available only when Type is Unique. Type the symbol name for the specified dimension to which the data is imported. Alternatively, you can use the symbol selector. **Note:** You can also use a variable to define the symbol
Map Location	This field is available only when Type is Map. Indicate the location of the map using one of the following values: • Internal — Indicates the map exists inside the database, created using the Mappings editor. For more information, see Maintaining mappings. • External — Indicates the map exists as a file, such as a text file on your local machine or network location. • Document — Indicates the map exists as a document within the Symbol Maps folder of the app.

Field	Description
Map	This field is available only when Type is Map. Do one of the following: • If you selected Internal as the Map Location, select a map from the list of available maps in the system. • If you selected External as the Map Location, type the path of the external map, or click the ellipsis button (...) to select the external map. • If you selected Document as the Map Location, select a map from the list of available maps in the app. **Note:** You can use a variable for the Mao field if the map location is Internal or External

7. For a single value, file-based export: in the Value Field table complete the following field:

Field	Description
Value field name	Specify the name of the field that contains the value. The field name is used to specify filters. The default value for this field is 'value'.
Value field number	Specify the field position at which the data value will be written using a number.
v26.2 Value field decimals	Specify the number of decimal places for exported numeric values. This setting controls the precision of the exported numbers. If not specified, the default is 9 decimal places.

8. If the data target contains multiple value fields, the Value Fields section is available. Complete the following fields:

Field	Description
Field Name	Optionally, enter the name of the field. If the field name is not specified it will be set to Field with the Field Position appended. The field name is used to specify filters.
Field Position	Specify the field position of the value field using a positive integer, where the field position is the corresponding column in the data source containing the data. For example, if the value is exported to the third column in the target file, type 3.
Symbol	Type the name of a symbol from the dimension indicated in Multiple Value Fields for Dimension for the value in the corresponding field position. Alternatively, you can use the symbol selector. **Note:** You can use a variable for the Symbol field.

Note: You can delete a value field by selecting a value field row and clicking Delete.

9. In the Data Options table, complete the following fields:

Field	Description
Enclose string values in quotes	Select this option to enclose string values double quotes in the target file.
Reverse value for credit accounts	Select this option to reverse the sign for numeric values for accounts with Credit as the Balance Type. For more information on the balance type in the ACCOUNTS dimension, see "Working with dimensions" in the *Longview Application Administrator Guide*. The default is No.
Decimal character	Specify the decimal character for the data import app using one of the following values: ■ " . " — The period is the decimal character. ■ " , " — The comma is the decimal character. ■ "Regional" — Use the regional character for the decimal. The default is the regional character for decimal.
Allow duplicate mappings	This option applies only when at least one dimension Type is Map. Select this option to indicate that duplicate mappings are permitted. If multiple mappings are found, the system does the following for each type of mapping: ■ Exact — All duplicate Exact mappings are used so that the same value is imported to multiple data cells. If there are duplicate Exact and Wildcard or Range mappings, the Exact mapping is used, and the Wildcard or Range mapping is ignored. ■ Wildcard — If no Exact mappings are found, the value is imported to the first Wildcard or Range mapping listed. ■ Range — If no Exact mappings are found, the value is imported to the first Wildcard or Range mapping listed. If you do not select this option, duplicate mappings are not permitted. If multiple mappings are found, the system reports an error for each mapping. The default is No.

Field	Description
Handling for duplicate records	Specify the way duplicate records are handled for the data import app using one of the following values: ■ Output all — Specifies that each duplicate record will be written to the target file. ■ Error — Specifies that duplicate records should not be allowed. The first entry is submitted. If a duplicate record is encountered, the system reports an error. ■ Sum values — Specifies that duplicate records should be allowed and the value of all such records summed. If a duplicate record is encountered that contains text (versus numeric) data, it is treated as an invalid/error record. The default is Error.
Number of errors before processing stops	Specify the maximum number of error records to permit before stopping the import process. The default is 0.
Header option	Specify whether to create a header in the target file, using one the following values: ■ None — Specifies that no header will be created and the first record in the file is a data record. ■ Auto — Specifies that a single header record is created in the target file. The header will contain the description of the dimension for each field that is a dimension, and ◦ "Value" for the value field in a single value export. ◦ The description of each symbol in a multiple value export. ■ Custom — Specifies that the header records will be manually entered in the Custom header field.
Custom header	When the header option is set to Custom, enter the header records in the form: "header record," "...", "header record." Each header record is output on a new line in the target file.

10. You can optionally specify filters to apply when exporting data. For each filter type an expression using the following guidelines:

    - The expression can be joined together by two or more expressions using AND or OR grouped by brackets, as appropriate.
    - Each expression uses field names, operators, and values as follows: FieldName EQ|NE|LE|LT|GE|GT Value. If Value is a string, enclose the string in double quotation marks. • You can optionally include wildcard characters (? and *) and symbols for Value.

      Note: In this case do not enclose Value in double quotation marks.

- You can optionally use symbols for operators, such as ==, !=, <=, <, >=, and >.
- You must use Field# for FieldName to refer to a specific field on which you want to filter. For example, if Products is the fourth column in your target file and you want to filter Products to import only the Products_Default symbol, use Field4 == "Products_Default" as the expression.

> **Note:** For more information on creating expressions, see “Using conditional operators” and “Search” in the *Longview Developer’s Guide*.

Specifying Export Specs - Script

An Export Spec (.lvexp) is used to export data to an external file.

For more information, see “Developing Longview ImportSpecs, ExportSpecs, and external Maps” in the *Longview Developer’s Guide*.

For more information on using the code editor, see Using the code editor.

> **Note:** If you would like to use a UI instead of a code editor to create the Export Spec, see Specifying Data Area Export Specs.

Specifying Forms

A Form (.lvfrm) defines the layout, format, and controls to display forms for user input.

For more information, see “Designing Longview Forms” in the *Longview Developer’s Guide*.

For more information on using the code editor, see Using the code editor.

Specifying HTML Documents

HTML files can be used to create any custom HTML pages, input forms or instructions for users.

For more information, see “Designing HTML pages” in the *Longview Developer’s Guide*.

Specifying Data Area Import Specs

Data area import specs (.lvimp) can be run from a procedure to import data to a data area. The source data may be contained in an external file, or a query run via an ODBC connection. The source data may contain a single column of values or multiple columns of values.

> **Note:** If you would like to use a code editor instead of a UI to create the Data Area Import Spec, see Specifying Import Specs - Script.

To create a data area import spec:

1. Right-click on one of the Data Area Import Specs folders and select New Document.
2. For the Name, enter the name of your Import Spec.

3. Fill in the data source.

   - For a single value, file-based import: in the File Options table, complete the following fields:

Field	Description
Source file	Specify the path and name of the data source.
Field delimiter	Specify whether the character used to separate fields in the data source is the brace ({), comma (,), semicolon (;), pipe (|), or regional delimiter for comma separated values (, (regional)). Alternatively, you can type a field delimiter of your choice. The default is the regional delimiter for comma separated values (, (regional)).
Number of header records to filter	Specify the number of header rows to filter on import. If you leave the value as the default value of 0, the system assumes your data source starts with data records with no header rows.
Number of footer records to filter	Specify the number of footer rows to filter on import. If you leave the value as the default value of 0, the system assumes your data source ends with data records with no footer rows.

   - For a single and multiple value, ODBC-based import, in the ODBC Parameters table, complete the following fields:

Field	Description
ODBC connection string	Specify the connect string required to connect to the ODBC source.
SQL query specification	Select the document that contains the SQL query to execute.
SQL query batch size	Enter the number of records that should be returned in the query batch. Default is 1000.

4. For a multiple value, file-based import, and ODBC-based import: in the Values Dimension table, complete the following fields:

Field	Description
Dimension corresponding to values	Specify the dimension for which there are multiple value fields.

5. The Dimension Definitions section is used to specify how each dimension is processed during import. For Dimension Definitions, complete the following:

   - Schedule: This field displays only if your system is configured to use schedules and you have access to schedules.

a. The default of None indicates the import spec is for base data only.

b. Specify the schedule to use for the import spec, and the schedule dimensions appear at the bottom of the list of Dimensions.

- For each dimension, complete the following fields:

Field	Description
Dimension	The name of the dimension.
Type	Specify the type of relationship between dimension symbols and field values in the data source using one of the following values: • Consecutive — The values encountered in the data source are placed consecutively in the dimension starting at the indicated symbol, moving in priority order through the indicated symbol's siblings. **Note:** Consecutive is designed to work with a symbol existing in only one hierarchy. If the symbol belongs to more than one hierarchy, the behavior used to determine the siblings is undefined and may be affected by hierarchy reorganizations and imports/exports. You can specify only one dimension as Consecutive. • Map — The values encountered in the data source are mapped to dimension symbols as specified in the indicated map. • Match — The values encountered in the data source are matched to the dimension symbols exactly. • Unique — All values encountered in the data source are placed in the specified dimension symbol. If you select this option, you can use the variables you defined on the Variables page for the specified symbol. **Note:** For a multiple value import spec, the dimension you indicated has Values for the Type field and cannot be changed.
Field Name	This field is available only when Type is Map or Match. Optionally, enter the name of the field. If the field name is not specified it will be set to Field with the Field Position appended. The field name is used to specify filters.
Field Position	This field is available only when Type is Map or Match. Specify the field position corresponding to the dimension using a positive integer. For example, for the Accounts dimension, type 3 if the account is in the third column of the data source.

Field	Description
Symbol	This field is available only when Type is Consecutive or Unique. Type the symbol name for the specified dimension to which the data is imported. Alternatively, you can use the symbol selector. **Note:** You can also use a variable to define the symbol.
Map Location	This field is available only when Type is Map. Indicate the location of the map using one of the following values: • Document — Indicates the map exists as a document within the Symbol Maps folder of the app. • Internal — Indicates the map exists inside the database, created using the Mappings editor. For more information, see Maintaining mappings. • External — Indicates the map exists as a file, such as a text file on your local machine or network location. • Document — Indicates the map exists as a document within the Symbol Maps folder of the app.
Map	This field is available only when Type is Map. Do one of the following: • If you selected Internal as the Map Location, select a map from the list of available maps in the system. • If you selected External as the Map Location, type the path of the external map, or click the ellipsis button (...) to select the external map. • If you selected Document as the Map Location, select a map from the list of available maps in the app. **Note:** You can use a variable for the Map field if the map location is Internal or External.

6. For a single value, file-based import, and ODBC-based import: in the Value Field table complete the following field:

Field	Description
Value field name	Specify the name of the field that contains the value. The field name is used to specify filters. The default value for this field is 'value'.

Field	Description
Value field number	Specify the field position at which the data value will be written using a number.

7. If the data source contains multiple value fields, the Value Fields section is available. Complete the following fields:

Field	Description
Field Name	Optionally, enter the name of the field. If the field name is not specified it will be set to Field with the Field Position appended. The field name is used to specify filters.
Field Position	Specify the field position of the value field using a positive integer, where the field position is the corresponding column in the data source containing the data. For example, if the value is in the third column in the import file, type 3.
Symbol	Type the name of a symbol from the dimension indicated in Multiple Value Fields for Dimension for the value in the corresponding field position. Alternatively, you can use the symbol selector. You can use a variable for the symbol field

Note: You can delete a value field by selecting a value field row and clicking Delete.

8. In the Data Options table, complete the following fields:

Field	Description
String values enclosed in quotes	Select this option if string values are enclosed in double quotes in the source file.
Reverse value for credit accounts	Select this option to reverse the sign for numeric values for accounts with Credit as the Balance Type. For more information on the balance type in the ACCOUNTS dimension, see “Working with dimensions” in the *Longview Application Administrator Guide*. The default is cleared.
Decimal character	Specify the decimal character for the data import app using one of the following values: ■ “ . ” — The period is the decimal character. ■ “ , ” — The comma is the decimal character. ■ “ . (regional) ” — Use the regional character for the decimal. The default is the regional character for decimal “ . (regional) ”.

Field	Description
Allow duplicate mappings	This option applies only when at least one dimension Type is Map. Select this option to indicate that duplicate mappings are permitted. If multiple mappings are found, the system does the following for each type of mapping: ■ Exact — All duplicate Exact mappings are used so that the same value is imported to multiple data cells. If there are duplicate Exact and Wildcard or Range mappings, the Exact mapping is used, and the Wildcard or Range mapping is ignored. ■ Wildcard — If no Exact mappings are found, the value is imported to the first Wildcard or Range mapping listed. ■ Range — If no Exact mappings are found, the value is imported to the first Wildcard or Range mapping listed. ■ If you do not select this option, duplicate mappings are not permitted. If multiple mappings are found, the system reports an error for each mapping. The default is No.
Handling for duplicate records	Specify the way duplicate records are handled for the data import app using one of the following values: ■ Error — Specifies that duplicate records should not be allowed. The first entry is submitted. If a duplicate record is encountered, the system reports an error. ■ Sum values — Specifies that duplicate records should be allowed and the value of all such records summed. If a duplicate record is encountered that contains text (versus numeric) data, it is treated as an invalid/error record. ■ Use first record — Specifies that the first entry of duplicate records should be used. If a duplicate record is encountered, the system does not report an error. ■ Use last record — Specifies that the last entry of duplicate records should be used. If a duplicate record is encountered, the system does not report an error. The default is Error
Number of errors before processing stops	Specify the maximum number of error records to permit before stopping the import process. The default is 0.

9. If the data source is a file, the Filters section is available. For each filter type an expression using the following guidelines:

- The expression can be joined together by two or more expressions using AND or OR grouped by brackets, as appropriate.
- Each expression uses field names, operators, and values as follows: FieldName EQ|NE|LE|LT|GE|GT Value. If Value is a string, enclose the string in double quotation marks.
- You can optionally include wildcard characters (? and *) and symbols for Value.

  Note: In this case do not enclose Value in double quotation marks.

- You can optionally use symbols for operators, such as ==, !=, <=, <, >=, and >.
- You must use Field# for FieldName to refer to a specific field on which you want to filter. For example, if Products is the fourth column in your import file and you want to filter Products to import only the Products_Default symbol, use Field4 == "Products_Default" as the expression.

  Note: For more information on creating expressions, see “Using conditional operators” and “Search” in the *Longview Developer’s Guide*.

Specifying Import Specs - Script

Import specs (.lvimp) can be run from a procedure and used to import data from an external file.

For more information, see “Developing Longview ImportSpecs, ExportSpecs, and external Maps” in the *Longview Developer’s Guide*.

For more information on using the code editor, see Using the code editor.

Specifying Models

Models (.lvmod) support the logic and calculations for the Longview App.

For more information, see “Developing Longview Models” in the *Longview Developer’s Guide*.

For more information on using the code editor, see Using the code editor.

Specifying Model Subroutines

Model Subroutines (.lvsub) can be used to avoid repeating common calculations in multiple Longview model documents. You can also create a subroutine when you have to create a complex calculation.

For more information, see “Subroutine Documents and Call function” in the Longview.

For more information on using the code editor, see Using the code editor.

Specifying Procedures

Procedures (.lvpro) define a sequence of commands to perform activities.

For more information, see “Developing Longview Procedures” in the *Longview Developer’s Guide*.

For more information on using the code editor, see Using the code editor.

Specifying Symbol Maps

Symbol Maps(.lvmap) specify instructions when the symbol names in your data source do not match the symbol names in your Longview database.

For more information, see “Developing Longview ImportSpecs, ExportSpecs, and external Maps” in the *Longview Developer’s Guide*.

For more information on using the code editor, see Using the code editor.

Specifying Persistent Table Definitions

Persistent table definitions allow you to define the content from the persistent app table that is downloaded to the in-memory DataTable object.

For more information on this file, see “Defining DataTable objects” in the *Longview Developer’s Guide*.

For more information on using the code editor, see Using the code editor.

> **Note:** If you would like to use a code editor instead of a UI to create the Persistent Table Definition, see Specifying Table Definitions - Script.

To create a table definition:

1. Right-click on Persistent Table Definition and select New Document.
2. For the Name, enter the name of your Table Definition.
3. In the columns section the table and columns are defined.

a. For the table name, select the table to use. The list includes all app tables defined in the system.

b. If you pick a table based on a view it will display an information icon indicating that the table will be read-only.

c. After selecting a table, the columns in the table will be displayed. All columns will be selected by default. Also, the columns that define the primary key will be indicated with a key icon. If some or all the primary key columns are not included, the table will be read only.

d. You may add or remove temporary columns using the Add Column and Remove Column buttons.

> **Note:** Temporary columns are added to the table for use with an app but are not submitted to the database.

e. For each column complete the following fields:

Field	Description
Include	Check each column to be included in the query. Use the check box in the header to select all columns.
Column	For persistent table columns, displays the name of the column. For temporary columns, enter the name of the column.
Nullable	Indicates whether the column can be saved with no value specified. If any non-nullable columns are excluded the table will be read-only.

Field	Description
Type	For persistent table columns, displays the data type of the column. For temporary columns, select the data type for the column. If you select Symbol[Dimension], you will be prompted to select the dimension that applies. If you select Symbol[DimensionColumnName], you will be prompted to select a column (of type Dimension).
Values (String)	For String columns allows you to optionally define a list of values the user can select from. Enter a | delimited list of values. Optionally enter a | delimited list of display values by placing a semi-colon after the list of values. **Note:** You can also use variables to define the list of values and display values.
Values (Symbol)	For Symbol columns allows you to limit the symbols available. Enter a | delimited list of symbol specifications. You can also use a form to select the symbol specifications by clicking the search icon. **Note:** You can also use a variable to define the list of symbol specifications
Values (User)	For User columns, allows you to limit the users available for selection. Enter (or select) a | delimited list of user and group names. **Note:** You can also use a variable to define the list of users and groups.
Values (UserList)	For UserList columns, allows you to limit the users available for selection. Enter (or select) a | delimited list of user and group names. **Note:** You can also use a variable to define the list of users and groups.
Allow Override	For String columns with Values defined, indicates whether the user can enter a value that is not in the list of values.
Min/Max (Date)	For date columns, allows you to set a valid range of selectable dates. **Note:** The interface only supports explicit dates. To specify the valid date range using variables use the Additional Configuration section to define the date range using the SetColumn function.
Default (Boolean)	For Boolean columns, allows you to indicate if the default value for the column is checked.

Field	Description
Default (Dimension)	For Dimension columns, allows you to set he default dimension to be selected when a new row is added. Default is no default selection.

4. Use additional configuration to:
   a. Define sort order.
   b. Define any filters (SymbolFilter, UserFilter, or Where)

Specifying Virtual Table Definitions

Virtual table definitions allow you to define a table to be created as an in-memory DataTable object.

For more information on using the code editor, see Using the code editor.

Note: If you would like to use a code editor instead of a UI to create the Virtual Table Definition, see Specifying Table Definitions – Script.

To create a table definition:

1. Right-click on Virtual Table Definition and select New Document.
2. For the Name, enter the name of your Table Definition.
3. In the columns section the table and columns are defined.
   a. You may add or remove columns using the Add Column and Remove Column buttons.
   b. For each column complete the following fields:

Field	Description
Include	Check each column to be included in the query. Use the check box in the header to select all columns.
Column	Enter the name of the column.
Nullable	Not selectable. Columns in a virtual table are nullable by default.
Type	Select the data type for the column. If you select Symbol[Dimension], you will be prompted to select the dimension that applies. If you select Symbol[DimensionColumnName], you will be prompted to select a column (of type Dimension).

Field	Description
Values (String)	For String columns allows you to optionally define a list of values the user can select from. Enter a | delimited list of values. Optionally enter a | delimited list of display values by placing a semi-colon after the list of values. **Note:** You can also use variables to define the list of values and display values.
Values (Symbol)	For Symbol columns allows you to limit the symbols available. Enter a | delimited list of symbol specifications. You can also use a form to select the symbol specifications by clicking the search icon. **Note:** You can also use a variable to define the list of symbol specifications.
Values (User)	For User columns, allows you to limit the users available for selection. Enter (or select) a | delimited list of user and group names. **Note:** You can also use a variable to define the list of users and groups.
Values (UserList)	For UserList columns, allows you to limit the users available for selection. Enter (or select) a | delimited list of user and group names. **Note:** You can also use a variable to define the list of users and groups.
Allow Override	For String columns with Values defined, indicates whether the user can enter a value that is not in the list of values.
Min/Max (Date)	For date columns, allows you to set a valid range of selectable dates. **Note:** The interface only supports explicit dates. To specify the valid date range using variables use the Additional Configuration section to define the date range using the SetColumn function.
Default (Boolean)	For Boolean columns, allows you to indicate if the default value for the column is checked.
Default (Dimension)	For Dimension columns, allows you to set he default dimension to be selected when a new row is added. Default is no default selection.

4. Use additional configuration to:
    a. Define sort order.

Specifying Table Definitions - Script

Table Definitions Script allows you to create a Table Definition document (.lvdtd) using a code editor. Table definitions allow you to define the content from the App table that is downloaded to the in-memory DataTable object.

For more information on this file, see “Defining DataTable objects” in the *Longview Developer’s Guide*.

For more information on using the code editor, see Using the code editor.

Note: If you would like to use a UI instead of a code editor to create the Table Definition, see Specifying Persistent Table Definitions or Specifying Virtual Table Definitions.

Specifying Table Views

Use Table Views to define the way in which a DataTable object is rendered as a Table. There is a separate folder for views based on persistent and virtual data tables. The only difference is the list of available table definitions in the Columns section.

Note: If you would like to use a code editor instead of a UI to create the Table View, see Specifying Table Views - Script.

Fill in the table view sections:

1. For the Columns, select the Table Definition that will be used with the view. The columns selected in the Data Table Definition appear in the table. For each column complete the following fields:

Field	Description
Include	Check each column to be included in the query. Use the check box in the header to select all columns.
Column	Displays the name of the column.
Type	Displays the data type of the column.
Description	Allows you to enter a description for the column to be displayed in the view
Width	Allows you to specify the width of the column in the view.
Protect	Check to protect the column from input.
Decimals (Number)	Allows you to specify the number of decimals displayed for numeric values.
Enabled Criteria	Allows you to specify criteria which allows the user to modify the value of the column in each row of the table.

2. In the Parameters table, complete the following fields:

Field	Desciption
Default number of decimals for numeric values	Specifies the number of decimals to display for numeric numbers (0-9). **Note:** The number of decimals for a specific column can be overridden using the ColumnDecimals function in the View Column Settings section.
Default column width	Specifies the width of each column in the view. **Note:** The width for a specific column can be overridden using the ColumnWidth function in the View Column Settings section.
Apply users' layout preferences	Select Yes to allow the user's modifications to the view to be saved and applied when the app is accessed. Select No to lock the settings of the initial view each time the app is accessed. **Note:** The user can modify the view while using the app, but the next time it is executed the view will revert to the one saved in the app.
Use parentheses for negative numbers	Select Yes to display negative numbers with parentheses. Select No to display negative numbers with a negative sign
Use thousands separator for numbers	Select Yes to display a locale-specific thousands separator based on the operating system setting. Select No to not display a thousand separator.

3. In the Values Pane Options, complete the following fields

Field	Description
Enable values pane	Select whether the user will be able to open the values pane within the view. Default is Yes.
Values pane title	Enter text to appear in the title area of the values pane. Default is Row Details
Hide values pane initially	Select whether the values panel in initially hidden, if enabled. Default is Yes.

4. Use Additional Configuration to further configure the view. Additional Configuration is a code editor that allows you to enter view functions to be included in the configuration of the view.

   For more information on the code editor, see Using the code editor.

   Typical use of View Column Settings in this app would be to:

   - Sort the rows in the view by one or more columns using the SortOrder function.
   - Adding summary rows to the view using the SummaryRow function.

- Adding dynamic calculations to the table using the DynamicProcedure function.
- Adding a custom editor for a column using the EditProcedure function.

Specifying Table Views - Script

Table Views Script allows you to create a Table View document (.lvtvw) using a code editor. Table Views define the way in which a DataTable object is rendered as a Table.

For more information on this file, see “Creating Table View definition files” in the *Longview Developer’s Guide*.

For more information on using the code editor, see Using the code editor.

> **Note:** For persistent tables only, if you would like to use a UI instead of a code editor to create the Table View, see Specifying Table Views.

Specifying UIs

UIs allow you to provide users with tabbed Apps, which can include Data Grids, Tables, and Calendars and are invoked using the Show UI command..

Tabbed Apps allow users to view different views within one window. Users can click tabs to change their view of the data in the system.

When a user switches from one tab to another, the following actions occur in the following order:

- the LVS_BLUR_ID system variable is populated with the label of the tab that has lost focus
- the LVS_FOCUS_ID system variable is populated with the label of the tab that has gained focus
- the BLUR command runs
- the FOCUS command runs

Each tab has its own toolbar; however, if a user clicks a toolbar action that has a SHOW RETURN command associated with it or the Close system action, the entire App closes.

Syntax

```
LAYOUTTYPE Tabbed

ObjectType, ObjectName, [ViewType,] "DefinitionFile", "Label", "OnFocus",
OnBlur"[,"TabColor"]
```

where:

- ObjectType can be one of the following:
  - DATAAREA — Use this value to add a Data Grid to a tab.
  - DATATABLE — Use this value to add a Table to a tab.

  Note To display multiple tabs, add multiple ObjectType statements. You can create tabbed Apps with any combination of Data Grids and Tables.

- ObjectName can be one of the following:
  - For Data Grids — The name of the Data Area to display in the tab.
  - For Tables or Calendars — The name of the DataTable to display in the tab.

- ViewType applies only if you specified DATATABLE for the ObjectType and can be one of the following:
  - TABLE specifies to render the DATATABLE object as a Table in the tab.
  - CALENDAR specifies to render the DATATABLE object as a Calendar in the tab.

- DefinitionFile is one of the following:
  - For Data Grids — The Data View definition (.lvdvw file) to use when displaying the Data Area.
  - For Tables — The Table View definition (.lvtvw file) to use when displaying the DataTable.
  - For Calendars — The Calendar View definition (.lvcvw file) to use when displaying the DataTable.

- Label is the label to display on the tab.

- OnFocus specifies the command to run when a user focuses on this tab. If you do not want to specify an OnFocus command, use two double quotation marks ("").

- OnBlur specifies the command to run when a user removes the focus from this tab. This parameter is optional. If you do not want to specify an OnBlur command, use two double quotation marks ("").

- TabColor is the background color for the tab. Use any Windows media brush color. If you do not specify a tab color, the tab is white.

Note: Windows media brush colors are case-sensitive.

Example: Data Grids Only

```
LayoutType TABBED
DataArea daNIISource, "NII\NII_CDs.lvdvw", "Loan Details", "", ""
```

```
DataArea daNIISource2, "NII\NII_View_CDDetail.lvdvw", "Cash Flow Details",
"", "Run PROCEDURE calcCF.lvpro"

DataArea daNIISource, "NII\NII_View_CDRollForward.lvdvw", "Cash Flow
Rollforward", "Run PROCEDURE cfRollForward.lvpro", ""

DataArea daNIICashFlow, "NII\NII_View_CDCashFlow.lvdvw", "Financial
Summary", "", ""

DataArea daNIIYieldSource, "NII\NII_View_CDYield.lvdvw", "Yield Curve", "",
"", "DarkOrchid"
```

Example: Tables Only

```
LayoutType TABBED

DataTable Salaries, TABLE, "Salary Reports\ExecutiveCompensation.lvtvw",
"Executive Compensation Report", "", "",

DataTable Salaries, TABLE, "Salary Reports\TopFiftySalaries.lvtvw", "Top 50
Salaries", "", "","DarkOrchid"
```

Example: Calendars Only

```
LayoutType TABBED

DataTable, ImportantDates, CALENDAR, "ImportantDates.lvcvw", "Important
Dates", "", "","Lavender"
```

Example: A Combination Of The Above

```
LayoutType TABBED

DataArea AccountsReceivable, "AccountsRec\ARNewYorkState.lvdvw", "Accounts
Receivable - NY, "", "Run PROCEDURE calcAR.lvpro" "Khaki"

DataTable AccountsRecList, TABLE, "AccountsRec\NYTopTwentyFive.lvtvw", "Top
25 Accounts Receivable", "", "", "Honeydew"

DataTable, ImportantDates, CALENDAR, "ImportantDates.lvcvw", "Important
Dates", "", "", "Lavender"
```

To specify the position of a new window:

Use the WindowPosition function in a .lvui file to specify the window position for new windows invoked by the SHOW UI command.

Syntax

```
WindowPosition x,y|TOPLEFT|CENTER
```

where:

- x,y specifies the coordinates, in pixels, at which to render the top-left corner of new windows. Coordinates are relative to the top-left of the screen for main windows or to the parent window for subwindows.
- TOPLEFT specifies to open new windows and subwindows with the top-left corner at the top left of the screen for new windows or parent window for new subwindows.
- CENTER specifies to open new windows or subwindows centered relative to the screen for new windows or relative to the parent window for new subwindows.

Examples

```
WindowPosition TOPLEFT
WindowPosition 50,100
```

For details on using the code editor, see Code editor. For more information on the Show UI command, see “Show UI” in the Developer’s Guide.

Specifying Supporting Files

Supporting Files can be any additional files you require within the free form app. For example, supplemental text (.txt) files, or sql query files (.sql).

Creating A Data View Input App Using Free Form App

You may come across cases where the data view input app template is insufficient for a particular case. You can use the free form app to replicate the standard functionality of the data view input and extend it to meet your specific needs. The steps provided here will guide you on incorporating the standard functionality of an input app but is intended to be used by developers experienced with developing apps in Longview.

To create a data view input app using free form app:

1. Create a new free form app.
2. Open the Execute Procedure.
3. Type “TEMP” in the code editor. A menu appears.
4. Select **TEMPLATE_DataViewInput_Execute** from the menu. Standard procedure code for the execute procedure of a data view input app is inserted.
5. Modify the execute procedure, as necessary.
   a. If the app will use any input tools (adjust values, copy across, quick input, or spreads), in the Initialize section, add:

```
Run PROCEDURE "VIEW\InitTools.lvpro"
```

b. If the app will use standard validation functionality, in the initialize section, add:

```
Run PROCEDURE "VALD\Init.lvpro"
```

c. To specify dimensions to include in the validation results set the variable VALD_DimensionList to the names of the dimensions to include. Enclose in double quotes and separate each dimension name with a pipe character. This is normally only used in cases where there are slice dimensions or multiple nested dimensions in the view and it may not be obvious to the user exactly where the validation error exists.

d. You can add standard symbol selection to the app or incorporate it into a custom form. Symbol selection occurs before the “Create lock” section. The remaining commands in Execute Procedure should only be executed if the user clicks OK. See Generating a symbol selection form or Generating symbol selection controls.

e. If the input app will use input related schedules (comments, file attachments or line-item detail), add UseInputRelatedSchedules commands as appropriate at the beginning of the Create Lock section.

f. If the pattern spread is made available to the user in the app patterns must be downloaded. You can use the procedure template Patterns_Retrieve to insert the required procedure code. This must be inserted before the Show DATAAREA command.

> **Note:** The “Determine pattern periods from data area” section should be repeated for each data area created for which pattern spread will be enabled.

g. Further modifications are noted in the following steps, depending on whether the app is a single data view or tabbed data view input.

6. Create the query specs. Repeat for each query spec:
   a. Right-click on **Data Area Definitions** to create a new document.
   b. Change the name of the document.

   > **Note:** For a single view input name, the document Main to minimize the code changes required from the template execute procedure.

   c. In the code block, specify the symbols to query for each dimension.

   > **Note:** Depending on the purpose of the app you might use some query specs to set locks as well.

7. Modify the “Prepare for input” section of the execute procedure to create all the query areas defined.
8. Create the lock specs. Repeat for each lock spec:
   a. Right-click on **Data Area Definitions** to create a new document.
   b. Change the name of the document.
   c. In the code block, specify the symbols to lock in each dimension.

9. Create the lock procedure:

> **Note:** The lock procedure is called from the code library procedure CORE\OnLock.lvpro which performs the standard control around locking.

   a. Right-click on **Procedures** to create a new document.
   b. Change the name of the document to “Lock.”
   c. In the code block enter all the Create Lock commands required to lock data areas in the app.

> **Note:** Note Use the Create Lock template in the code editor to quickly add the commands.

10. Create the refresh procedure:

> **Note:** The refresh procedure is called from the code library procedure VIEW\OnRefresh.lvpro which performs the standard control around refreshing data areas. This procedure is called prior to the view and whenever the user clicks Refresh.

   a. Right-click on **Procedures** to create a new document.
   b. Change the name of the document to "Refresh".
   c. In the code block enter all the Download commands required to download data into the data areas.
   d. If there are any calculations to perform after download, add Run MODEL commands as required after the Download command.
   e. After the Run MODEL commands use the Set DATAAREA command to reset the data changed indicator to false, so the user is not prompted to save changes, if the app is immediately closed.

> **Note:** Add any models referenced in the refresh procedure by right-clicking on Models to create a document.

11. Create the save procedure:

> **Note:** The save procedure is called from the code library procedure VIEW\OnSave.lvpro which performs the standard control around saving.

   a. Right-click on **Procedures** to create a new document.
   b. Change the name of the document to "Save".
   c. In the code block enter the Upload command required to submit data from the data area to the database.
   d. If there are any calculations to perform prior to submitting, add Run MODEL commands as required before the Upload command.
   e. If there are any calculations to perform after submitting, add Run MODEL commands as required after the Upload command.

f. To display the user submission details after upload, add the following code:

Sample code

```
If Count(LVS_BATCH_IDS)
For EACH  iDVI_BatchID IN LVS_BATCH_IDS
Set VARIABLE VIEW_BatchIDList = ListAppend(VIEW_BatchIDList, $iDVI_
BatchID$)
Next
END If
```

> **Note:** Add any models referenced in the save procedure by right-clicking on Models to create a document.

12. If using validations in the app, create the validate procedure:

> **Note:** The validate procedure is called from the code library procedure VIEW\OnSave.lvpro, when the user clicks Submit and the code library procedure VALD\ViewResults.lvpro, when the user clicks Validate.

a. Right-click on **Procedures** to create a new document.
b. Change the name of the document to Validate.
c. In the code block enter the commands to execute the validations. See Writing validations for details.

13. If allowing the use of the standard import tool in the app, create the import procedure:

> **Note:** The import procedure is called from the code library procedure VIEW\OnImport.lvpro which performs the standard control around importing data.

a. Right-click on **Procedures** to create a new document
b. Change the name of the document to Import.
c. In the code block insert the DataViewInput_Import template.

14. Create the views. Repeat for each view required:
a. Right-click on **Data Views** to create a new document.
b. Change the name of the document.

> **Note:** For a single view input name, the document Main to minimize the code changes required from the template execute procedure.

c. In the code block enter the functions to format the view.

> **Note:**
> - The code editor provides templates for a data view document which can be used to provide an outline of a view document.
> - The code editor also provides function templates for adding standard functionality available in the data view input app to your view. Add standard items such as Import and Quick Spread using Action templates.

d. For any models that need to be run as the user enters data, specify them using the DynamicModel function. Use one DynamicModel function for each model to be executed. The models will be executed in the order listed in the document.

e. If allowing use of the standard import tool, add the icon to the toolbar using the Action_Template_InputApp_Import in the code editor.

15. For a tabbed input with multiple data views:
    a. Right-click on **UIs** to create a new document.
    b. Change the name of the document.
    c. In the code block enter the DataArea keywords required to display each view in a tab.
    d. Change the Show DATAREA command in the Execute Procedure to a Show UI command.
16. At this point you have created a free form app that contains the standard functionality of a data view input app. Add any additional items to round out the app.

Writing validations

Validation checks are performed by testing a value in the data area against an expected value and providing a warning or error depending on whether the validation failure should prevent the user from submitting data.

Repeat for each validation:

1. Create a variable to store the validation result.

   > **Note:** You can use the same result variable for each validation.

2. Write an expression to determine the validation result. Store the result in the variable created in step 1.
3. Write a condition to check the validation result against the expected result.
4. If the validation result does not pass the check, Set the VALD_Result variable and run procedure VALD\AddResult.lvpro to add the validation failure result to the list of validation failures.

Example

```
Create VARIABLE val as NUM
```

```
Set VARIABLE val = Value("daMain", "BS", "$TIMEPERIODS$", "$ENTITIES$",
"GL", "SCENARIOS_Working", "$CURRENCIES$")

If $val$ != 0

Set VARIABLE VALD_Result = "ERROR|Balance sheet is out of balance|$val$"

Run PROCEDURE "VALD\AddResult.lvpro"

END If
```

Example (multiple entities)

```
Create VARIABLE val as NUM

For EACH entity in ENTITIES

Set VARIABLE val = Value("daMain", "BS", "$TIMEPERIODS$", "$entity$", "GL",
"SCENARIOS_Working", "$CURRENCIES$")

If $val$ != 0

Set VARIABLE VALD_Result = "ERROR|Balance sheet is out of
balance|$val$|$entity$"

Run PROCEDURE "VALD\AddResult.lvpro"

END If

Next
```

Setting value for variable VALD_Result

The VALD_Result variable is a string list variable, set as follows (separate each item with a pipe character “|”):

1. ERROR or WARNING to specify the validation failure level.
2. The validation failure message to display to the user.
3. The validation failure amount.
4. If validation dimensions are specified, the symbol for each dimension.

Creating Configuration Libraries

You can use Longview Designer to create configuration libraries.

Specifying properties

You can use Longview Designer to specify properties for a new configuration library.

For more information, see Specifying properties.

Specifying Calculated Journal Entry Specs

A Calculated Journal Entry Spec (.lvcje) is an ASCII file that specifies information about a calculated journal entry such as the journal entry type and description, as well as a line of detail for each adjustment that the calculated journal entry contains.

For more information, see “Developing Longview CalculatedJESpecs” in the *Longview Developer’s Guide*.

For more information on using the code editor, see Using the code editor

Specifying Data Area Definitions

Use Data Area Definitions to create a dataspec document (.lvdsp) used to define an area in the database. For each data area definition to be created, right-click on Data Area Definitions and select New Document.

Note: If you would like to use a code editor instead of a UI to create the Data Area Definition, see Specifying Data Area Definitions – Script.

1. For the Name, enter the name of your Data Area
2. In the Options table, complete the following field

Field	Description
Data Type	Specify the data to be queried using one of the following options: ■ ADJUSTED — Indicates that data queries include adjustments made via journal entries ■ UNADJUSTED — Indicates that data queries exclude adjustments made via journal entries Default is ADJUSTED

3. In the Data Area table, complete the following fields for each dimension:

Note: The Spec and Level fields are not applicable if a symbol contains a variable.

Field	Description
Symbols	Type a symbol name. Alternatively, you can use the symbol selector. To add more than one symbol for a dimension, select a row containing the dimension for which you want to add another symbol and click Symbol. To move a symbol up or down within a dimension, click Move Up or Move Down.
Spec	Indicate the symbol specification using one of the following values: ■ All — Includes all symbols within the selected symbol’s hierarchy. ■ Leaf — Includes only leaf symbols within the selected symbol’s hierarchy. ■ Parent — Includes only parent symbols within the selected symbol’s hierarchy. ■ Root and Parent — Includes only root and parent symbols within the selected symbol’s hierarchy.

Field	Description
Level	Specify the number of levels down from the symbol to be included in the data area.

4. Use Additional Configuration to specify any other Data Area Definition features you wish to use. Additional Configuration is a code editor that allows you to enter data area definition functions to be included in the data area.

   For more information on the code editor, see Using the code editor.

   Typical use of Additional Configuration in a data area definition would be to

   - Add an attribute filter using the AttributeFilter function
   - Add a schedule to the using the Schedule function
   - Add a temporary symbol using the TempSym function

Specifying Data Area Definitions - Script

Data Area Definitions - Script allows you to create a DataSpec document (.lvdsp) using a code editor. This defines an area in the database.

For more information, see “Developing Longview DataSpecs” in the *Longview Developer’s Guide*.

For more information on using the code editor, see Using the code editor.

> **Note:** If you would like to use a UI instead of a code editor to create the Data Area Definition, see Specifying Data Area Definitions.

Specifying Data Area Views

Use Data Area Views to create a data view document (.lvdvw) used to specify how the data areas will be displayed to the user. For each data area view to be created, right-click on Data Area Views and select New Document.

> **Note:** If you would like to use a code editor instead of a UI to create the Data Area Views, see Specifying Data Area Views - Script.

Name

Specify the name of the data area view

Setting view orientation

Use orientation to determine how each dimension is presented to the user. At least one dimension must be in the rows and columns orientation. Multiple dimensions in rows or columns are nested outer to inner, based on their left to right position.

Use the following table to determine the way dimensions display in the view.

Dimension orientation	View appearance
Columns or Rows	The left-most dimension listed in the orientation section is the outermost dimension in the view.
Slice	Each dimension in the slice dimension is presented to the user as a symbol selector. Dimensions are presented in the view in the order that they appear in the Slice orientation.
Fixed	The view opens displaying fixed symbols in dimension order on the Fixed Symbols tab. Dimensions appear on the Fixed Symbols tab in their natural order regardless of the order listed in the Fixed orientation.

To modify dimension orientation:

1. In the Orientation section for the dimension, you want to reorient, select the dimension.
2. Drag and drop the selected dimension into the desired position.

Specifying symbols

Use symbol selection to specify the symbols that are displayed in the view.

In the Symbol Selection table, complete the following fields for each dimension: The Spec and Level fields are not applicable if a symbol contains a variable.

Field	Description
Symbols	Type a symbol name. Alternatively, you can use the symbol selector. To add more than one symbol for a dimension, select a row containing the dimension for which you want to add another symbol and click Symbol. To move a symbol up or down within a dimension, click Move Up or Move Down.
Spec	Indicate the symbol specification using one of the following values: ■ Root and Parent — Includes only root and parent symbols within the selected symbol's hierarchy. ■ All — Includes all symbols within the selected symbol's hierarchy. ■ Leaf — Includes only leaf symbols within the selected symbol's hierarchy. ■ Parent — Includes only parent symbols within the selected symbol's hierarchy. ■ Root and Parent — Includes only root and parent symbols within the selected symbol's hierarchy.
Level	Specify the number of levels down from the symbol to be included in the data area.
Levels to Expand	For dimensions in the rows and column orientation only, you can specify the number of levels to expand the hierarchy of each symbol included.

Specifying dimension display settings

Use dimension display settings to specify how symbols are displayed in the view. In the Dimension Display Settings table, complete the following fields for each dimension:

> **Note:** Show Name and Show Description only affect dimensions in the Rows or Columns orientation.

Field	Description
Show Name	Check this to display the name of the symbol in the view. If both Show Name and Show Description are selected the symbol name and description will be displayed separated by a hyphen for symbols in the dimension.
Show Description	Check this to display the description of the symbol in the view. If both Show Name and Show Description are selected the symbol name and description will be displayed separated by a hyphen for symbols in the dimension.
Fit to Title	Check this to make the view auto-size to fit symbol titles. For Rows this affects the width of the title. For Columns this can only be selected for the inner dimension and affects the width of the column.
Column Width	If Fit to Title is not selected, enter the width in pixels.
Wrap	If Fit to Title is not selected, check this to wrap titles onto new lines if the text does not fit in the specified width.

Specifying number formatting

Use Number Formatting to specify how numbers are displayed in the view.

In the Number Formatting table complete the following fields:

Field	Description
Default decimals	Enter the number of decimals to display for each numeric value. Generally, this should be from 0 to 9. This can be overridden for a specific symbol using the SymbolDecimals function in the Additional Configuration section.
Use parentheses for negative numbers	Select Yes to display negative numbers with parentheses. Select No to display negative numbers with a negative sign.
Use thousands separator for numbers	Select Yes to display a locale-specific thousands separator based on the operating system setting. Select No to not display a thousands separator.

Specifying options

Use Options to specify additional view settings.

In the Options table complete the following fields:

Field	Description
Allow dynamic rollup of input	Select Yes to have parent totals automatically calculated as the user enters numeric values.

Customizing the data view

Use Additional Configuration to specify any other view features you wish to use in the data view. Additional Configuration is a code editor that allows you to enter view functions to be included in the configuration of the view.

For more information on the code editor, see Using the code editor.

Typical use of Additional Configuration in this app would be to

- Add dynamic calculations to the view using the DynamicModel function
- Add protected areas to the view using the Protect function
- Add conditional style to the view using the ConditionalStyle function
- Restrict input to numeric values or text values using the NumericInputOnly and/or TextInputOnly functions
- Hiding specific symbols in the view using the Hide function
- Overriding the decimals displayed for certain symbols using the SymbolDecimals function
- Adding separators between symbols using the InsertSeparator function

Specifying Data Area Views - Script

Data Area Views - Script allows you to create a Data View document (.lvdsp) using a code editor. A Data View document defines the way in which the data area is rendered as a Data Grid.

For more information, see “Designing Longview Data Grids” in the *Longview Developer’s Guide*.

For more information on using the code editor, see Using the code editor.

> **Note:** If you would like to use a UI instead of a code editor to create the Data Area Views, see Specifying Data Area Views.”

Specifying Data Area Export Specs

Data area export specs (.lvexp) can be run from a procedure to export data to a file. The target file may contain a single column of values or multiple columns of values.

> **Note:** If you would like to use a code editor instead of a UI to create the Data Area Export Spec, see Specifying Export Specs – Script.”

To create a data area export spec:

1. Right-click on one of the Data Area Export Specs folders and select New Document.
2. For the Name, enter the name of your Export Spec.
3. Fill in the data target
4. In the File Options table, complete the following fields:

Field	Description
Target File	Specify the path and name of the data target.
Field Delimiter	Specify whether the character used to separate fields in the data source is the brace ({), comma (,), semicolon (;), pipe (|), or regional delimiter for comma separated values (Regional). Alternatively, you can type a field delimiter of your choice. The default is the regional delimiter for comma separated values.

5. For a multiple value, file-based export: in the Values Dimension table complete the following field:

Field	Description
Dimension corresponding to values	Specify the dimension for which there are multiple value fields.

6. The Dimension Definitions section is used to specify how each dimension is processed during export. For Dimension Definitions, complete the following:
   - Schedule: This field displays only if your system is configured to use schedules and you have access to schedules.
     a. The default of None indicates the export spec is for base data only.
     b. Specify the schedule to use for the import spec, and the schedule dimensions appear at the bottom of the list of Dimensions.
   - For each dimension, complete the following fields:

Field	Description
Dimension	The name of the dimension.

Field	Description
Type	Specify the type of relationship between dimension symbols and field values in the data source using one of the following values: • Map — The dimension symbols are mapped to values as specified in the indicated map • Match — The dimension symbols are written out with their names. • Unique — Values for the symbol specified are exported, but the symbol name is not written in the target file. **Note:** For a multiple values export spec, the dimension you indicated has Values for the Type field and cannot be changed.
Field Name	This field is available only when Type is Map or Match. Optionally, enter the name of the field. If the field name is not specified it will be set to Field with the Field Position appended. The field name is used to specify filters.
Field Position	This field is available only when Type is Map or Match. Specify the field position to assign the dimension to using a positive integer. For example, for the Accounts dimension, type 3 if the account will be the third column of the data target.
Symbol	This field is available only when Type is Unique. Type the symbol name for the specified dimension to which the data is imported. Alternatively, you can use the symbol selector. **Note:** You can also use a variable to define the symbol
Map Location	This field is available only when Type is Map. Indicate the location of the map using one of the following values: • Internal — Indicates the map exists inside the database, created using the Mappings editor. For more information, see Maintaining mappings. • External — Indicates the map exists as a file, such as a text file on your local machine or network location. • Document — Indicates the map exists as a document within the Symbol Maps folder of the app.

Field	Description
Map	This field is available only when Type is Map. Do one of the following: • If you selected Internal as the Map Location, select a map from the list of available maps in the system. • If you selected External as the Map Location, type the path of the external map, or click the ellipsis button (...) to select the external map. • If you selected Document as the Map Location, select a map from the list of available maps in the app. **Note:** You can use a variable for the Mao field if the map location is Internal or External

7. For a single value, file-based export: in the Value Field table complete the following field:

Field	Description
Value field name	Specify the name of the field that contains the value. The field name is used to specify filters. The default value for this field is 'value'.
Value Field Number	Specify the field position at which the data value will be written using a number.
v26.2 Value field decimals	Specify the number of decimal places for exported numeric values. This setting controls the precision of the exported numbers. If not specified, the default is 9 decimal places.

8. If the data target contains multiple value fields, the Value Fields section is available. Complete the following fields:

Field	Description
Field Position	Specify the field position of the value field using a positive integer, where the field position is the corresponding column in the data source containing the data. For example, if the value is exported to the third column in the target file, type 3.
Symbol	Type the name of a symbol from the dimension indicated in Multiple Value Fields for Dimension for the value in the corresponding field position. Alternatively, you can use the symbol selector. **Note:** You can use a variable for the Symbol field.

Note: You can delete a value field by selecting a value field row and clicking Delete.

9. In the Data Options table, complete the following fields:

Field	Description
Enclose string values in quotes	Select this option to enclose string values double quotes in the target file.
Reverse value for credit accounts	Select this option to reverse the sign for numeric values for accounts with Credit as the Balance Type. For more information on the balance type in the ACCOUNTS dimension, see "Working with dimensions" in the *Longview Application Administrator Guide*. The default is No.
Decimal character	Specify the decimal character for the data import app using one of the following values: ■ " . " — The period is the decimal character. ■ " , " — The comma is the decimal character. ■ "Regional" — Use the regional character for the decimal. The default is the regional character for decimal.
Allow duplicate mappings	This option applies only when at least one dimension Type is Map. Select this option to indicate that duplicate mappings are permitted. If multiple mappings are found, the system does the following for each type of mapping: ■ Exact — All duplicate Exact mappings are used so that the same value is imported to multiple data cells. If there are duplicate Exact and Wildcard or Range mappings, the Exact mapping is used, and the Wildcard or Range mapping is ignored. ■ Wildcard — If no Exact mappings are found, the value is imported to the first Wildcard or Range mapping listed. ■ Range — If no Exact mappings are found, the value is imported to the first Wildcard or Range mapping listed. If you do not select this option, duplicate mappings are not permitted. If multiple mappings are found, the system reports an error for each mapping. The default is No.

Field	Description
Handling for duplicate records	Specify the way duplicate records are handled for the data import app using one of the following values: ■ Output all — Specifies that each duplicate record will be written to the target file. ■ Error — Specifies that duplicate records should not be allowed. The first entry is submitted. If a duplicate record is encountered, the system reports an error. ■ Sum values — Specifies that duplicate records should be allowed and the value of all such records summed. If a duplicate record is encountered that contains text (versus numeric) data, it is treated as an invalid/error record. The default is Error.
Number of errors before processing stops	Specify the maximum number of error records to permit before stopping the import process. The default is 0.
Header option	Specify whether to create a header in the target file, using one the following values: ■ None — Specifies that no header will be created and the first record in the file is a data record. ■ Auto — Specifies that a single header record is created in the target file. The header will contain the description of the dimension for each field that is a dimension, and ◦ "Value" for the value field in a single value export. ◦ The description of each symbol in a multiple value export. ■ Custom — Specifies that the header records will be manually entered in the Custom header field.
Custom header	When the header option is set to Custom, enter the header records in the form: "header record," "...", "header record." Each header record is output on a new line in the target file.

10. You can optionally specify filters to apply when exporting data. For each filter type an expression using the following guidelines:

    - The expression can be joined together by two or more expressions using AND or OR grouped by brackets, as appropriate.
    - Each expression uses field names, operators, and values as follows: FieldName EQ|NE|LE|LT|GE|GT Value. If Value is a string, enclose the string in double quotation marks. • You can optionally include wildcard characters (? and *) and symbols for Value.

      Note: In this case do not enclose Value in double quotation marks.

- You can optionally use symbols for operators, such as ==, !=, <=, <, >=, and >.
- You must use Field# for FieldName to refer to a specific field on which you want to filter. For example, if Products is the fourth column in your target file and you want to filter Products to import only the Products_Default symbol, use Field4 == "Products_Default" as the expression.

> **Note:** For more information on creating expressions, see “Using conditional operators” and “Search” in the *Longview Developer’s Guide*.

Specifying Export Specs - Script

An Export Spec (.lvexp) is used to export data to an external file.

For more information, see “Developing Longview ImportSpecs, ExportSpecs, and external Maps” in the *Longview Developer’s Guide*.

For more information on using the code editor, see Using the code editor.

> **Note:** If you would like to use a UI instead of a code editor to create the Export Spec, see Specifying Data Area Export Specs.

> **Note:** If you are using an Export Spec, you must run EXP\Init.lvpro in the Execute Procedure to initialize variables.

Specifying Forms

A Form (.lvfrm) defines the layout, format, and controls to display forms for user input.

For more information, see “Designing Longview Forms” in the *Longview Developer’s Guide*.

For more information on using the code editor, see Using the code editor.

Specifying HTML Documents

HTML files can be used to create any custom HTML pages, input forms or instructions for users.

For more information, see “Designing HTML pages” in the *Longview Developer’s Guide*.

Specifying Data Area Import Specs

Data area import specs (.lvimp) can be run from a procedure to import data to a data area. The source data may be contained in an external file, or a query run via an ODBC connection. The source data may contain a single column of values or multiple columns of values.

> **Note:** If you would like to use a code editor instead of a UI to create the Data Area Import Spec, see Specifying Import Specs - Script.

To create a data area import spec:

1. Right-click on one of the Data Area Import Specs folders and select New Document.
2. For the Name, enter the name of your Import Spec.

3. Fill in the data source.

   - For a single value, file-based import: in the File Options table, complete the following fields:

Field	Description
Source file	Specify the path and name of the data source.
Field delimiter	Specify whether the character used to separate fields in the data source is the brace ({), comma (,), semicolon (;), pipe (|), or regional delimiter for comma separated values (, (regional)). Alternatively, you can type a field delimiter of your choice. The default is the regional delimiter for comma separated values (, (regional)).
Number of header records to filter	Specify the number of header rows to filter on import. If you leave the value as the default value of 0, the system assumes your data source starts with data records with no header rows.
Number of footer records to filter	Specify the number of footer rows to filter on import. If you leave the value as the default value of 0, the system assumes your data source ends with data records with no footer rows.

   - For a single and multiple value, ODBC-based import, in the ODBC Parameters table, complete the following fields:

Field	Description
ODBC connection string	Specify the connect string required to connect to the ODBC source.
SQL query specification	Select the document that contains the SQL query to execute.
SQL query batch size	Enter the number of records that should be returned in the query batch. Default is 1000.

4. For a multiple value, file-based import, and ODBC-based import: in the Values Dimension table, complete the following fields:

Field	Description
Dimension corresponding to values	Specify the dimension for which there are multiple value fields.

5. The Dimension Definitions section is used to specify how each dimension is processed during import. For Dimension Definitions, complete the following:

   - Schedule: This field displays only if your system is configured to use schedules and you have access to schedules.

a. The default of None indicates the import spec is for base data only.

b. Specify the schedule to use for the import spec, and the schedule dimensions appear at the bottom of the list of Dimensions.

- For each dimension, complete the following fields:

Field	Description
Dimension	The name of the dimension.
Type	Specify the type of relationship between dimension symbols and field values in the data source using one of the following values: • Consecutive — The values encountered in the data source are placed consecutively in the dimension starting at the indicated symbol, moving in priority order through the indicated symbol's siblings. **Note:** Consecutive is designed to work with a symbol existing in only one hierarchy. If the symbol belongs to more than one hierarchy, the behavior used to determine the siblings is undefined and may be affected by hierarchy reorganizations and imports/exports. You can specify only one dimension as Consecutive. • Map — The values encountered in the data source are mapped to dimension symbols as specified in the indicated map. • Match — The values encountered in the data source are matched to the dimension symbols exactly. • Unique — All values encountered in the data source are placed in the specified dimension symbol. If you select this option, you can use the variables you defined on the Variables page for the specified symbol. **Note:** For a multiple value import spec, the dimension you indicated has Values for the Type field and cannot be changed.
Field Name	This field is available only when Type is Map or Match. Optionally, enter the name of the field. If the field name is not specified it will be set to Field with the Field Position appended. The field name is used to specify filters.
Field Position	This field is available only when Type is Map or Match. Specify the field position corresponding to the dimension using a positive integer. For example, for the Accounts dimension, type 3 if the account is in the third column of the data source.

Field	Description
Symbol	This field is available only when Type is Consecutive or Unique. Type the symbol name for the specified dimension to which the data is imported. Alternatively, you can use the symbol selector. **Note:** You can also use a variable to define the symbol.
Map Location	This field is available only when Type is Map. Indicate the location of the map using one of the following values: • Document — Indicates the map exists as a document within the Symbol Maps folder of the app. • Internal — Indicates the map exists inside the database, created using the Mappings editor. For more information, see Maintaining mappings. • External — Indicates the map exists as a file, such as a text file on your local machine or network location. • Document — Indicates the map exists as a document within the Symbol Maps folder of the app.
Map	This field is available only when Type is Map. Do one of the following: • If you selected Internal as the Map Location, select a map from the list of available maps in the system. • If you selected External as the Map Location, type the path of the external map, or click the ellipsis button (...) to select the external map. • If you selected Document as the Map Location, select a map from the list of available maps in the app. **Note:** You can use a variable for the Map field if the map location is Internal or External.

6. For a single value, file-based import, and ODBC-based import: in the Value Field table complete the following field:

Field	Description
Value field name	Specify the name of the field that contains the value. The field name is used to specify filters. The default value for this field is ‘value ’.
Value field number	Specify the field position at which the data value will be written using a number.

7. If the data source contains multiple value fields, the Value Fields section is available. Complete the following fields:

Field	Description
Field Name	Optionally, enter the name of the field. If the field name is not specified it will be set to Field with the Field Position appended. The field name is used to specify filters.
Field Position	Specify the field position of the value field using a positive integer, where the field position is the corresponding column in the data source containing the data. For example, if the value is in the third column in the import file, type 3.
Symbol	Type the name of a symbol from the dimension indicated in Multiple Value Fields for Dimension for the value in the corresponding field position. Alternatively, you can use the symbol selector. You can use a variable for the symbol field

Note: You can delete a value field by selecting a value field row and clicking Delete.

8. In the Data Options table, complete the following fields:

Field	Description
String values enclosed in quotes	Select this option if string values are enclosed in double quotes in the source file.
Reverse value for credit accounts	Select this option to reverse the sign for numeric values for accounts with Credit as the Balance Type. For more information on the balance type in the ACCOUNTS dimension, see “Working with dimensions” in the *Longview Application Administrator Guide*. The default is cleared.
Decimal character	Specify the decimal character for the data import app using one of the following values: ■ “ . ” — The period is the decimal character. ■ “ , ” — The comma is the decimal character. ■ “ . (regional) ” — Use the regional character for the decimal. The default is the regional character for decimal “ . (regional) ”.

Field	Description
Allow duplicate mappings	This option applies only when at least one dimension Type is Map. Select this option to indicate that duplicate mappings are permitted. If multiple mappings are found, the system does the following for each type of mapping: ■ Exact — All duplicate Exact mappings are used so that the same value is imported to multiple data cells. If there are duplicate Exact and Wildcard or Range mappings, the Exact mapping is used, and the Wildcard or Range mapping is ignored. ■ Wildcard — If no Exact mappings are found, the value is imported to the first Wildcard or Range mapping listed. ■ Range — If no Exact mappings are found, the value is imported to the first Wildcard or Range mapping listed. ■ If you do not select this option, duplicate mappings are not permitted. If multiple mappings are found, the system reports an error for each mapping. The default is No.
Handling for duplicate records	Specify the way duplicate records are handled for the data import app using one of the following values: ■ Error — Specifies that duplicate records should not be allowed. The first entry is submitted. If a duplicate record is encountered, the system reports an error. ■ Sum values — Specifies that duplicate records should be allowed and the value of all such records summed. If a duplicate record is encountered that contains text (versus numeric) data, it is treated as an invalid/error record. ■ Use first record — Specifies that the first entry of duplicate records should be used. If a duplicate record is encountered, the system does not report an error. ■ Use last record — Specifies that the last entry of duplicate records should be used. If a duplicate record is encountered, the system does not report an error. The default is Error
Number of errors before processing stops	Specify the maximum number of error records to permit before stopping the import process. The default is 0.

9. If the data source is a file, the Filters section is available. For each filter type an expression using the following guidelines:

- The expression can be joined together by two or more expressions using AND or and grouped by brackets, as appropriate.
- Each expression uses field names, operators, and values as follows: FieldName EQ|NE|LE|LT|GE|GT Value. If Value is a string, enclose the string in double quotation marks.
- You can optionally include wildcard characters (? and *) and symbols for Value.

  Note: In this case do not enclose Value in double quotation marks.

- You can optionally use symbols for operators, such as ==, !=, <=, <, >=, and >.
- You must use Field# for FieldName to refer to a specific field on which you want to filter. For example, if Products is the fourth column in your import file and you want to filter Products to import only the Products_Default symbol, use Field4 == "Products_Default" as the expression.

  Note: For more information on creating expressions, see “Using conditional operators” and “Search” in the *Longview Developer’s Guide*.

Specifying Import Specs - Script

Import specs (.lvimp) can be run from a procedure and used to import data from an external file.

For more information, see “Developing Longview ImportSpecs, ExportSpecs, and external Maps” in the *Longview Developer’s Guide*.

For more information on using the code editor, see Using the code editor.

Specifying Models

Models (.lvmod) support the logic and calculations for the Longview App.

For more information, see “Developing Longview Models” in the *Longview Developer’s Guide*.

For more information on using the code editor, see Using the code editor.

Specifying Model Subroutines

Model Subroutines (.lvsub) can be used to avoid repeating common calculations in multiple Longview model documents. You can also create a subroutine when you must create a complex calculation.

For more information, see “Subroutine Documents and Call function” in the Longview.

For more information on using the code editor, see Using the code editor.

Specifying Procedures

Procedures (.lvpro) define a sequence of commands to perform activities.

For more information, see Developing Longview Procedures.

For more information on using the code editor, see Using the code editor.

Specifying Symbol Maps

Symbol Maps(.lvmap) specify instructions when the symbol names in your data source do not match the symbol names in your Longview database.

For more information, see “Developing Longview ImportSpecs, ExportSpecs, and external Maps” in the *Longview Developer’s Guide*.

For more information on using the code editor, see Using the code editor.

Specifying Persistent Table Definitions

Persistent table definitions allow you to define the content from the persistent app table that is downloaded to the in-memory DataTable object.

For more information on this file, see “Defining DataTable objects” in the *Longview Developer’s Guide*.

For more information on using the code editor, see Using the code editor.

> **Note:** If you would like to use a code editor instead of a UI to create the Persistent Table Definition, see Specifying Table Definitions - Script.

To create a table definition:

Right-click on Persistent Table Definition and select New Document.

1. For the Name, enter the name of your Table Definition.
2. In the columns section the table and columns are defined.

a. For the table name, select the table to use. The list includes all app tables defined in the system.

b. If you pick a table based on a view it will display an information icon indicating that the table will be read-only.

c. After selecting a table, the columns in the table will be displayed. All columns will be selected by default. Also, the columns that define the primary key will be indicated with a key icon. If some or all the primary key columns are not included, the table will be read only.

d. You may add or remove temporary columns using the Add Column and Remove Column buttons.

> **Note:** Temporary columns are added to the table for use with an app but are not submitted to the database.

e. For each column complete the following fields:

Field	Description
Include	Check each column to be included in the query. Use the check box in the header to select all columns.
Column	For persistent table columns, displays the name of the column. For temporary columns, enter the name of the column.
Nullable	Indicates whether the column can be saved with no value specified. If any non-nullable columns are excluded the table will be read-only.

Field	Description
Type	For persistent table columns, displays the data type of the column. For temporary columns, select the data type for the column. If you select Symbol[Dimension], you will be prompted to select the dimension that applies. If you select Symbol[DimensionColumnName], you will be prompted to select a column (of type Dimension).
Values (String)	For String columns allows you to optionally define a list of values the user can select from. Enter a | delimited list of values. Optionally enter a | delimited list of display values by placing a semi-colon after the list of values. **Note:** You can also use variables to define the list of values and display values.
Values (Symbol)	For Symbol columns allows you to limit the symbols available. Enter a | delimited list of symbol specifications. You can also use a form to select the symbol specifications by clicking the search icon. **Note:** You can also use a variable to define the list of symbol specifications
Values (User)	For User columns, allows you to limit the users available for selection. Enter (or select) a | delimited list of user and group names. **Note:** You can also use a variable to define the list of users and groups.
Values (UserList)	For UserList columns, allows you to limit the users available for selection. Enter (or select) a | delimited list of user and group names. **Note:** You can also use a variable to define the list of users and groups.
Allow Override	For String columns with Values defined, indicates whether the user can enter a value that is not in the list of values.
Min/Max (Date)	For date columns, allows you to set a valid range of selectable dates. **Note:** The interface only supports explicit dates. To specify the valid date range using variables use the Additional Configuration section to define the date range using the SetColumn function.
Default (Boolean)	For Boolean columns, allows you to indicate if the default value for the column is checked.

Field	Description
Default (Dimension)	For Dimension columns, allows you to set he default dimension to be selected when a new row is added. Default is no default selection.

3. Use additional configuration to:
   a. Define sort order.
   b. Define any filters (SymbolFilter, UserFilter, or Where)

Specifying Virtual Table Definitions

Virtual table definitions allow you to define a table to be created as an in-memory DataTable object.

For more information on using the code editor, see Using the code editor.

Note: If you would like to use a code editor instead of a UI to create the Virtual Table Definition, see Specifying Table Definitions – Script.

To create a table definition:

1. Right-click on Virtual Table Definition and select New Document.
2. For the Name, enter the name of your Table Definition.
3. In the columns section the table and columns are defined.
   a. You may add or remove columns using the Add Column and Remove Column buttons.
   b. For each column complete the following fields:

Field	Description
Include	Check each column to be included in the query. Use the check box in the header to select all columns.
Column	Enter the name of the column.
Nullable	Not selectable. Columns in a virtual table are nullable by default.
Type	Select the data type for the column. If you select Symbol[Dimension], you will be prompted to select the dimension that applies. If you select Symbol[DimensionColumnName], you will be prompted to select a column (of type Dimension).

Field	Description
Values (String)	For String columns allows you to optionally define a list of values the user can select from. Enter a | delimited list of values. Optionally enter a | delimited list of display values by placing a semi-colon after the list of values. **Note:** You can also use variables to define the list of values and display values.
Values (Symbol)	For Symbol columns allows you to limit the symbols available. Enter a | delimited list of symbol specifications. You can also use a form to select the symbol specifications by clicking the search icon. **Note:** You can also use a variable to define the list of symbol specifications.
Values (User)	For User columns, allows you to limit the users available for selection. Enter (or select) a | delimited list of user and group names. **Note:** You can also use a variable to define the list of users and groups.
Values (UserList)	For UserList columns, allows you to limit the users available for selection. Enter (or select) a | delimited list of user and group names. **Note:** You can also use a variable to define the list of users and groups.
Allow Override	For String columns with Values defined, indicates whether the user can enter a value that is not in the list of values.
Min/Max (Date)	For date columns, allows you to set a valid range of selectable dates. **Note:** The interface only supports explicit dates. To specify the valid date range using variables use the Additional Configuration section to define the date range using the SetColumn function.
Default (Boolean)	For Boolean columns, allows you to indicate if the default value for the column is checked.
Default (Dimension)	For Dimension columns, allows you to set he default dimension to be selected when a new row is added. Default is no default selection.

4. Use additional configuration to:
    a. Define sort order.

Specifying Table Definitions - Script

Table Definitions Script allows you to create a Table Definition document (.lvdtd) using a code editor. Table definitions allow you to define the content from the App table that is downloaded to the in-memory DataTable object.

For more information on this file, see “Defining DataTable objects” in the *Longview Developer’s Guide*.

For more information on using the code editor, see Using the code editor.

Note: If you would like to use a UI instead of a code editor to create the Table Definition, see Specifying Persistent Table Definitions or Specifying Virtual Table Definitions.

Specifying Table Views

Use Table Views to define the way in which a DataTable object is rendered as a Table. There is a separate folder for views based on persistent and virtual data tables. The only difference is the list of available table definitions in the Columns section.

Note: If you would like to use a code editor instead of a UI to create the Table View, see Specifying Table Views - Script.

Fill in the table view sections:

1. For the Columns, select the Table Definition that will be used with the view. The columns selected in the Data Table Definition appear in the table. For each column complete the following fields:

Field	Description
Include	Check each column to be included in the query. Use the check box in the header to select all columns.
Column	Displays the name of the column.
Type	Displays the data type of the column.
Description	Allows you to enter a description for the column to be displayed in the view
Width	Allows you to specify the width of the column in the view.
Protect	Check to protect the column from input.
Decimals (Number)	Allows you to specify the number of decimals displayed for numeric values.
Enabled Criteria	Allows you to specify criteria which allows the user to modify the value of the column in each row of the table.

2. In the Parameters table, complete the following fields:

Field	Desciption
Default number of decimals for numeric values	Specifies the number of decimals to display for numeric numbers (0-9). **Note:** The number of decimals for a specific column can be overridden using the ColumnDecimals function in the View Column Settings section.
Default column width	Specifies the width of each column in the view. **Note:** The width for a specific column can be overridden using the ColumnWidth function in the View Column Settings section.
Apply users' layout preferences	Select Yes to allow the user's modifications to the view to be saved and applied when the app is accessed. Select No to lock the settings of the initial view each time the app is accessed. **Note:** The user can modify the view while using the app, but the next time it is executed the view will revert to the one saved in the app.
Use parentheses for negative numbers	Select Yes to display negative numbers with parentheses. Select No to display negative numbers with a negative sign
Use thousands separator for numbers	Select Yes to display a locale-specific thousands separator based on the operating system setting. Select No to not display a thousand separator.

3. In the Values Pane Options, complete the following fields

Field	Description
Enable values pane	Select whether the user will be able to open the values pane within the view. Default is Yes.
Values pane title	Enter text to appear in the title area of the values pane. Default is Row Details
Hide values pane initially	Select whether the values panel in initially hidden, if enabled. Default is Yes.

4. Use Additional Configuration to further configure the view. Additional Configuration is a code editor that allows you to enter view functions to be included in the configuration of the view.

   For more information on the code editor, see Using the code editor.

   Typical use of View Column Settings in this app would be to:

   - Sort the rows in the view by one or more columns using the SortOrder function.
   - Adding summary rows to the view using the SummaryRow function.

- Adding dynamic calculations to the table using the DynamicProcedure function.
- Adding a custom editor for a column using the EditProcedure function.

Specifying Table Views - Script

Table Views Script allows you to create a Table View document (.lvtvw) using a code editor. Table Views define the way in which a DataTable object is rendered as a Table.

For more information on this file, see “Creating Table View definition files” in the *Longview Developer’s Guide*.

For more information on using the code editor, see Using the code editor.

> **Note:** For persistent tables only, if you would like to use a UI instead of a code editor to create the Table View, see Specifying Table Views.

Specifying UIs

UI's define the behavior and format of Apps.

For more information, see “Formatting Longview Apps” in the *Longview Developer’s Guide*.

For more information on using the code editor, see Using the code editor.

Specifying Supporting Files

Supporting Files can be any additional files you require within the free form app. For example, supplemental text (.txt) files, or sql query files (.sql).

Maintaining Mappings

Users with the appropriate permissions can create, view, and modify maps and mappings to maintain system data. A map is a collection of mappings for a specified dimension. A mapping is a rule for translating an account code or other code in your source file to a Longview symbol name when importing data into the Longview database.

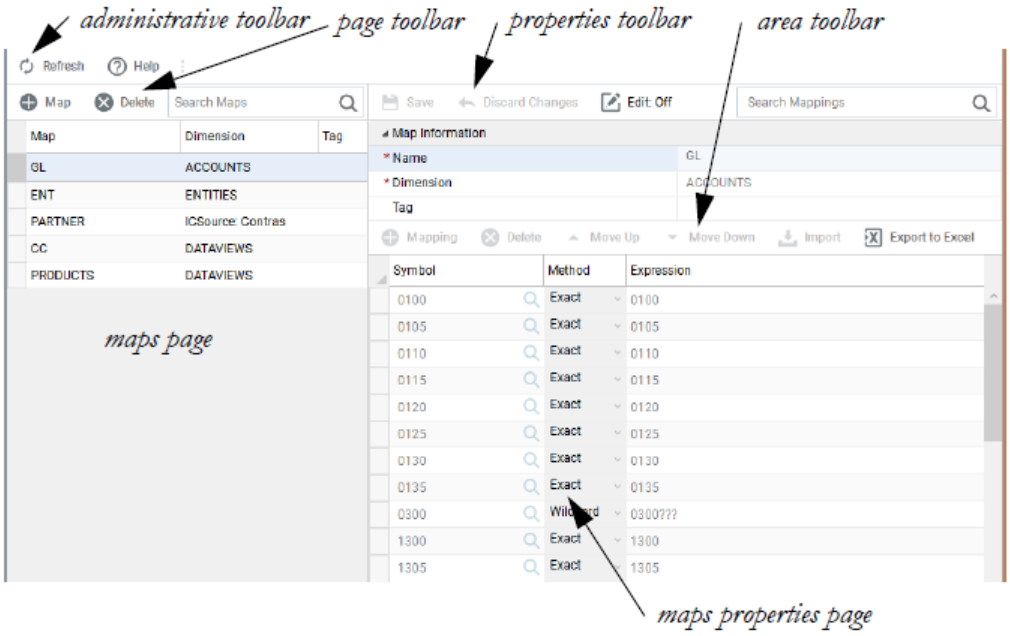
You can use maps created using the Mappings editor in data import apps in Longview Designer and in the Define function in Longview Application Framework.

For more information on data import apps, see the *Longview Designer Guide*.

For more information on the Define function, see the *Longview Developer's Guide*.

Understanding the Mappings editor interface

The Mappings editor contains a unique layout to maximize screen space and display as many mappings as possible.



Mappings editor

The Mappings editor is divided into the maps page and the map properties page.

All maps in the system are displayed in a list on the maps page. When you select a map, the map's properties and all mappings for the selected map are displayed in a list on map properties page.

The Mappings editor includes the administrative toolbar.

Administrative toolbar

The following table lists the button that appears on the administrative toolbar:

Button	Description
Refresh	Click the Refresh button to update all information in the current view to reflect any changes that any other users have made.

Maps page

The maps page contains a list of all maps that currently exist in your system. The list shows the name, dimension, and tag for each map.

The maps page includes the page toolbar.

Page toolbar

The following table lists the buttons that appear on the page toolbar:

Button	Description
Map	Click the Map button to add a new map to the system.
Delete	Click the Delete button to delete a selected map from the system.

Map properties page

The map properties page contains the map information area and the mappings area. The map information area contains the selected map's name, dimension, and tag. The mappings area contains a list of all mappings for a selected map. The map properties page contains the properties toolbar and the area toolbar.

Properties toolbar

The following table lists the buttons that appear on the properties toolbar:

Button	Description
Save	Click the Save button to save any changes you have made to the properties of the selected map.
Discard Changes	Click the Discard Changes button to remove any unsaved changes you have made to the properties of the selected mapping.
Edit	Click the Edit button to switch Edit mode On or Off. You must have Edit mode On to modify map properties including mappings.

Area toolbar

The following table lists the buttons that appear on the mappings area toolbar:

Button	Description
Mapping	Click the Mapping button to add a new mapping to the selected map.
Delete	Click the Delete button to delete one or more selected mapping from the selected map.
Move Up	Click the Move Up button to move a mapping up one row.

Button	Description
Move Down	Click the Move Down button to move a mapping down one row.
Import	Click the Import button to import mappings to the selected map.
Export to Excel	Click the Export to Excel button to export mappings from the selected map.

Working with maps

All maps in the system are displayed in a list on the Maps page. Depending on your authorizations and symbol access, you can view, add, modify, or delete maps.

Note: Mappings authorization is set by your Administrator

Accessing the Mappings editor

You can access the Mappings editor from the following locations, depending on the setup of your system, the products you have purchased, and your authorizations:

- Longview Dashboard — The Mappings editor is included with Longview Dashboard Designer as a system panel. Your System Administrator must add this panel to the appropriate Dashboard pages for you to access it.
- Longview Designer — The Mappings editor is available from the Administration category of the Designer navigation pane.
- Longview Tax — The Mappings editor is available from the Administration category of the Tax Provision navigation pane.

To access the Mappings editor from your Longview Dashboard:

1. On your Longview Dashboard page, click the Mappings link in the Mappings panel. The Mappings editor opens in a new window.

   Note: If this is the first time you are running the Mappings editor, you may have to complete additional steps. For more information, see the *Longview Dashboard User's Help*.

To access the Mappings editor from Longview Designer:

1. In the Designer navigation pane, click Administration.
2. Click Mappings. The Mappings editor opens in the workspace.

To access the Mappings editor from Tax Provision:

1. In the Tax Provision navigation pane, click Administration.
2. Expand System and click Mappings. The Mappings editor opens in the workspace.

Adding maps

If you have the appropriate authorization, you can use the Mappings editor to add a new map to the system.

To add a map:

1. Open the Mappings editor using the appropriate method. For more information, see Accessing the Mappings editor.
2. Click Map. A new row appears at the bottom of the maps list.
3. Complete the following fields:

Feild	Description
Name	Type a name for the map. The map name must be unique and can have a maximum of 255 characters, including spaces.
Dimension	Select the dimension to which the map applies. Accessible schedule dimensions also appear in the list.
Tag	Type a tag to assign to the map. Tag is optional and can have a maximum of 255 characters, including spaces.

4. Click Save. The map appears in the maps list.
5. To add mappings to the map, proceed to Adding mappings or Importing mappings.

Editing maps

If you have the appropriate authorization, you can use the Mappings editor to edit the properties of a map including the name, dimension, and tag. When Edit mode is On, the map is locked and other users cannot make changes.

To edit a map:

1. Open the Mappings editor using the appropriate method. For more information, see Accessing the Mappings editor.
2. In the maps list, select the map you want to edit, and click Edit. Edit mode switches to Edit: On.

   Caution: Changing the Dimension of a map removes all existing mappings.

3. Make the necessary changes and click Save. The map is saved.
4. If you are done editing the selected map, click Edit. Edit mode switches to Edit: Off. If you want to modify mappings, proceed to Adding mappings or Editing mappings.

Searching maps

You can use the Search Maps box to search for a map.

To search maps:

1. Open the Mappings editor using the appropriate method. For more information, see Accessing the Mappings editor.

2. Type in the Search Maps box. Matches, if any, appear highlighted yellow in the maps page.
3. If your search returned more than one result, click Next to navigate the search results.

Sorting maps

You can sort the list of maps by name, dimension, or tag.

To sort maps:

1. Open the Mappings editor using the appropriate method. For more information, see Accessing the Mappings editor.
2. On the maps page, click the row header of the field by which you want to sort maps. Maps appear sorted alphabetically under the selected field.

Deleting maps

If you have the appropriate authorization, you can use the Mappings editor to delete a map.

Caution: Deleting a map deletes all the associated mappings. You cannot undo the deletion of a map; you must re-create the map and its mappings.

To delete a map:

1. Open the Mappings editor using the appropriate method. For more information, see Accessing the Mappings editor.
2. In the maps list, select the map you want to delete, and click Delete. A confirmation dialog opens.
3. Click Delete. The map is removed from the system, including all associated mappings.

Working with mappings

All maps in the system are displayed in a list on the maps page. When you select a map, the map's properties and all mappings for the selected map are displayed on the map properties page. Depending on your authorizations and symbol access, you can view, add, modify, or delete mappings.

Note: Mapping authorization is set by your Administrator.

Understanding mapping methods and expressions

Maps contain instructions for importing data when the symbol names in your source file do not match the symbol names in the Longview database. A map contains a number of mappings, which specify methods for the way in which a source symbol should be mapped to a Longview database symbol using an expression.

You can create internal maps using the Mappings editor and external maps in Longview Application Framework.

For more information on a creating a sample external map, see "Sample External Map" in the *Longview Developer's Guide*. You can use one of three methods in a mapping:

- Exact
- Wildcard
- Range
- In

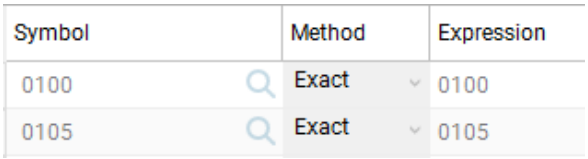
You can use all alphanumeric and ASCII characters in an expression except for double quotation marks (”). The following are the rules for each method:

Exact

Use this method to match the expression value exactly. For example, use the exact method to map a single source symbol name to a Longview database symbol name exactly as indicated in the expression.

Note: Special characters are matched exactly when used in an expression for the Exact method.

- Internal syntax



- External syntax

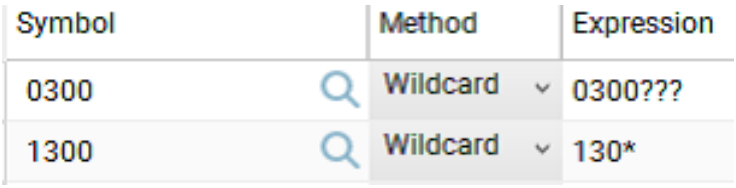
```
Map Cash, EXACT, 10000
Map AccountsReceivable, EXACT, 20000
```

Wildcard

Use this method to match the expression field partially by including wildcards in an expression. This method is case sensitive. Use the question mark (?) to specify any one character as wildcard. Use the asterisk (*) to specify zero or more characters as wildcards. You can use multiple question marks in an expression but only one asterisk. For example, use the wildcard method to import source symbols beginning with certain characters to a Longview database symbol name.

Note: If you use an asterisk, it must be the last character in the expression

- Internal syntax



Symbol	Method	Expression
0300	Wildcard	0300???
1300	Wildcard	130*

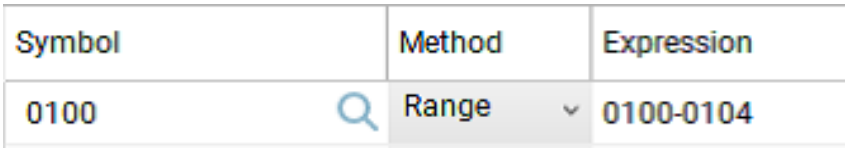
- External syntax

```
Map Cash, WILDCARD, 1000?
Map AccountsReceivable, WILDCARD, 200*
```

Range

Use this method to match a range of values, separated by a hyphen. This method is case sensitive. For example, use the range method when you have a range of source symbol names to import to a Longview database symbol name.

- Internal syntax



Symbol	Method	Expression
0100	Range	0100-0104

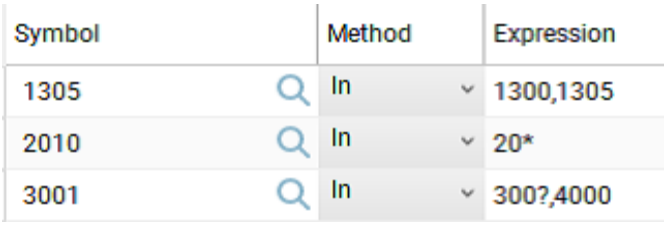
- External syntax

```
Map Cash, RANGE, 10000-10009
```

In

Use this method to match the expression value to one of several values. For example, use the IN method when you have a set of source symbol names to import to a Longview database symbol name. This method is case sensitive. You may also use the wildcard characters question mark (?) and asterisk (*) in the IN clause.

- Internal syntax



Symbol	Method	Expression
1305	In	1300,1305
2010	In	20*
3001	In	300?,4000

- External syntax

```
Map E12130, IN, "E12130,E11130, E11140, E12000"
Map E13100, IN, "E13*,E00000"
Map E11110, IN, "E11??0,E11111"
```

Understanding duplicate mappings

When you use the Mappings editor to add mappings, the map containing the mappings is used to import data using a data import app. The designer of a data import app can permit duplicate mappings using the Allow Duplicate Mappings setting in Longview Designer. If Allow Duplicate Mappings is not selected for a data import app, duplicate mappings are not permitted. If duplicate mappings are not permitted and multiple mappings are found, the system reports an error for each mapping.

In some cases, you may add duplicate mappings intentionally. You may want to, for example, translate an account code in your source file to multiple Longview symbols. This is classified as a duplicate mapping.

Note: Do not confuse duplicate mappings with duplicate records. A duplicate record occurs when you translate multiple account codes in your source file to one Longview symbol. You can use the Duplicate Records setting in Longview Designer to specify the way the system treats duplicate records.

Use the following table to review sample duplicate mappings for each mapping method and the way the system imports the data when Allow Duplicate Mappings is selected.

Method	Sample mappings in the Mappings editor	The system...
Exact	A1000 Exact 10020 A1001 Exact 10020 A1002 Exact 10020	uses all duplicate Exact mappings so that the same value is imported to multiple data cells. In this case, the value from 10020 is imported to A1000, A1001, and A1002.
Wildcard	A2000 Wildcard 112* A1000 Wildcard 11*	imports the first matching Wildcard or Range mapping listed.
Range	A3000 Range 13000-13009 A4000 Range 12000-13999	imports the first matching Wildcard or Range mapping listed.

Order

When Allow Duplicate Mappings is selected, the order of mappings in a map does not matter for Exact mappings. Consider the order of mappings contained in the following two maps, and the result:

Map A

```
A1000 Exact 10020
A1100 Wildcard 10*
```

Map B

```
A1100 Wildcard 10*
A1000 Exact 10020
```

In both cases, the system imports the value from 10020 to A1000.

Order matters for Wildcard and Range mappings. The system imports the value based on the first matching Wildcard or Range mapping listed. Consider the order of mappings in the following map for the source file account code 10020:

```
A3000 Range 10010-10030
A4000 Wildcard 100*
```

In this case, the value from 10020 is imported to A3000, not to A4000, because the specified Range mapping is listed first and includes 10020.

Adding mappings

If you have the appropriate authorization, you can use the Mappings editor to add mappings to a map.

Note: Typically, you should list Exact mappings first, followed by Wildcard and Range mappings.

To add a mapping:

1. Open the Mappings editor using the appropriate method. For more information, see Accessing the Mappings editor.
2. From the map list, select the map to which you want to add mappings. The list of mappings for the selected map appears in the maps properties page.
3. Click Edit. Edit mode switches to Edit: On.
4. Do one of the following:
   - To add a new mapping to the bottom of the list, click Mapping. The new mapping appears at the bottom of the mappings list.
   - To add a new mapping below a specific mapping, select a row and click Mapping. The new mapping appears below the selected row.

5. Complete the following fields:

Field	Description
Symbol	Type a symbol name or use the Symbol Selector to specify the symbol for the mapping.
Method	Select one of the following mapping methods: **Note:** If Allow Duplicate Mappings is selected for Data Import Apps, duplicate mappings are handled differently depending on the specified mapping method. For more information, see Understanding duplicate mappings. ■ Exact — Use this method to match the expression field value exactly. ■ Wildcard — Use this method to match the expression field partially by including the question mark (?) to specify any one character and/or the asterisk (*) to specify zero or more characters. You can use multiple question marks but only one asterisk in an expression. **Note:** If you use an asterisk, it must be the last character in an expression. ■ Range — Use this method to match a range of values, separated by a hyphen (-). **Note:** The start range and end range values must have the same number of characters
Expression	Type the expression for the symbol mapping. For more information on creating expressions, see Understanding mapping methods and expressions.

6. Click Save. The mapping appears in the mappings list.
7. If you are done adding mappings to the selected map, click Edit. Edit mode switches to Edit: Off.

Editing mappings

If you have the appropriate authorization, you can use the Mappings editor to edit a mapping.

Note: You can edit mappings only for symbols to which you have write access.

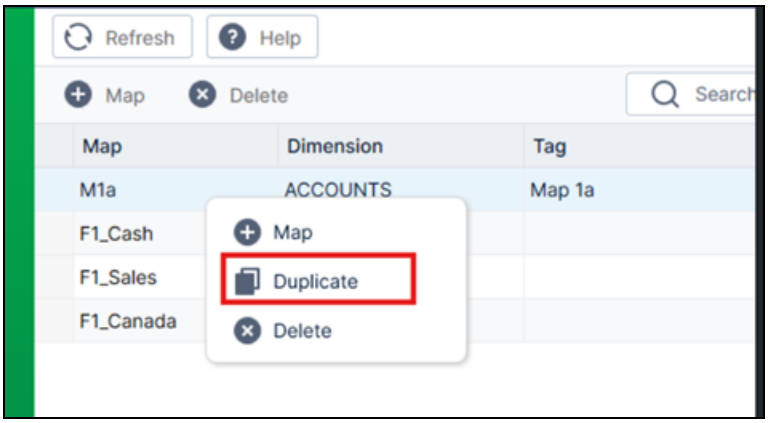
To edit a mapping:

1. Open the Mappings editor using the appropriate method. For more information, see Accessing the Mappings editor.
2. In the maps list, select the map containing the mapping you want to edit. The list of mappings for the selected map appears on the mappings page.

3. Click Edit. Edit mode switches to Edit: On.
4. Make the necessary changes and click Save. The mapping is saved.
5. If you are done editing mappings in the selected map, click Edit. Edit mode switches to Edit: Off.

v26.2 Duplicating a Mapping Table

The Duplicate feature enables administrators to quickly create a new mapping table by copying an existing one, saving time and ensuring consistency.



How to Duplicate a Mapping Table:

1. Navigate to the list of mapping tables and right-click the mapping you want to duplicate.
2. Select Duplicate from the context menu.
3. A new mapping row is created with an empty Name field, and the following fields are pre-populated :

- Dimension: Copied from the original mapping.
- Tag: Copied from the original mapping and can be modified.
- Mapping Data: All symbols, methods, and expressions are copied.

4. Enter a unique name in the Name field. The system ensures all mapping names are unique.
5. Click Save to create the new mapping with the copied data or Cancel to discard the operation.

Importing mappings

If you have the appropriate authorization, you can use the Mappings editor to import mappings. You can use a comma-delimited (.csv) file to import mappings into a map.

The .csv file must contain the following information on each line for each mapping:

```
Symbol, Method, Expression
```

To import mappings:

1. Open the Mappings editor using the appropriate method. For more information, see Accessing the Mappings editor.
2. From the maps list, select the map to which you want to import mappings.
3. Click Edit. Edit mode switches to Edit: On.
4. Click Import.

Note: v26.2 The system ignores the first row of a CSV file if it contains headers, and import succeeds with or without them.

Note: If mappings already exist, you will be prompted to choose how to handle them: i) Overwrite - which deletes all existing mappings and replaces them with the imported ones or ii) Append - which adds the imported mappings to the existing ones.

5. A dialog opens. Navigate to and select the .csv file containing the mappings you want to import, then click Open. The mappings will be appended or imported based on your previous selection.
6. If you are done importing mappings to the selected map, click Edit. Edit mode switches to Edit: Off.

v26.2 Exporting Mappings as CSV

If you have the appropriate authorization, you can use the Mappings editor to export mappings to a csv file.

Note: Any mappings with symbols you do not have access to are not exported.

To export a mapping table in CSV format:

1. Open the Mappings tool using the appropriate method. For more information, see Accessing the Mappings editor.
2. From the map list, select the map containing the mappings you want to export. The mappings list appears on the mappings page.
3. Click Export. A dialog opens.
4. Specify a name for the export file and click Save. The mapping table is exported with column headers included in the first row of the CSV file.

Exporting mappings to Microsoft Excel

If you have the appropriate authorization, you can use the Mappings editor to export mappings to Microsoft Excel.

Note: Any mappings with symbols you do not have access to are not exported.

To export mappings:

1. Open the Mappings tool using the appropriate method. For more information, see Accessing the Mappings editor.
2. From the map list, select the map containing the mappings you want to export. The mappings list appears on the mappings page.
3. Click Export to Excel. A dialog opens.
4. Specify a name for the export file and click Save. The mappings are exported to the specified file.

Searching mappings

You can use the Search Mappings box to search for a mapping.

To search mappings:

1. Open the Mappings editor using the appropriate method. For more information, see Accessing the Mappings editor.
2. From the map list, select the map containing the mappings you want to search. The mappings list for the selected map appears on the mappings page.
3. Type in the Search Mappings box. Matches, if any, appear highlighted yellow.
4. If your search returned more than one result, click Next to navigate the search results.

Reordering mappings

If you have the appropriate authorization, you can use the Mappings editor to change the order of mappings. The order of mappings is important because the system maps the source file symbol names to the database sequentially.

To reorder mappings:

1. Open the Mappings editor using the appropriate method.
   For more information, see Accessing the Mappings editor.
2. From the map list, select the map containing the mappings you want to reorder. The mappings list for the selected maps appears on the map properties page.
3. Click Edit. Edit mode switches to Edit: On.
4. Select the mapping you want to reorder, and then click either Move Up or Move Down. The mapping moves one row in the indicated direction.
5. If you want to continue to move the selected mapping, repeat step 4 until the mapping is in the desired location.
6. If you are done reordering mappings for the selected map, click Edit. Edit mode switches to Edit: Off.

Deleting mappings

If you have the appropriate authorization, you can use the Mappings editor to delete a mapping.

Note: You can delete mappings only for symbols to which you have write access.

To delete a mapping:

1. Open the Mappings editor using the appropriate method. For more information, see Accessing the Mappings editor.
2. In the maps list, select the map containing the mapping you want to delete. The list of mappings for the selected map appears on the map properties page.
3. Click Edit. Edit mode switches to Edit: On.
4. In the mappings list, select the mapping you want to delete, and click Delete. The mapping is deleted.

   Note: You can select multiple mappings by pressing and holding the Ctrl key on your keyboard while clicking rows. You can select multiple contiguous mappings using the Shift + click keystroke combination, or all mappings using the Ctrl + a keystroke combination.

5. If you are done deleting mappings in the selected map, click Edit. Edit mode switches to Edit: Off.

Process Maps Category

The process maps category contains the Manage Process Maps, Edit Process Maps, Import Process Maps, and the Export Process Maps app.

- Manage Process Maps: Use the process maps manager to create, edit and delete process maps.
- Edit Process Maps: Use the process maps editor to update process maps.
- Import Process Maps: Use the process map import to add or replace process maps with updated process maps for another system.
- Export Process Maps: Use the process map export to export one or more process maps to .json files for import in another system.

Manage Process Maps

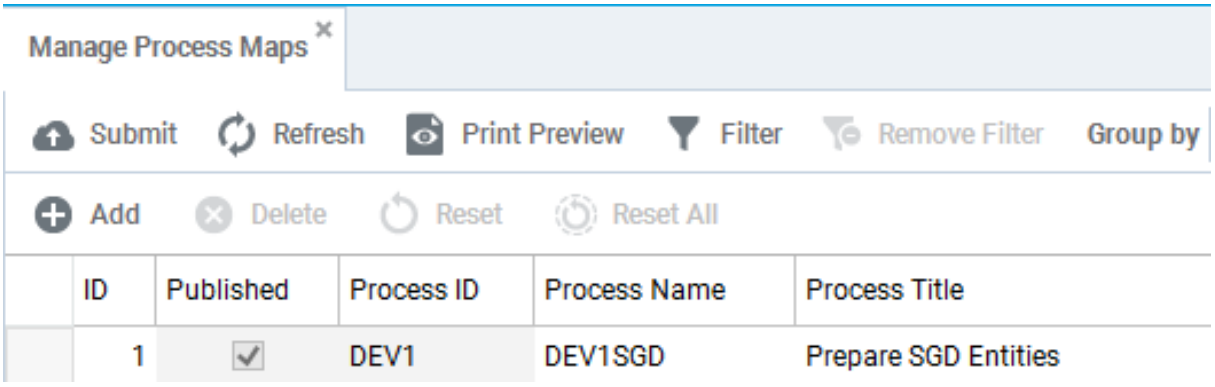
Manage Process Maps allows you to create, edit, delete, and rename process maps in your system.

To access manage process maps:

1. Select the Design module
2. Select the Process Maps category.
3. Click Mange Process Maps.
4. The Manage Process Maps table appears.

Managing process maps

The existing process maps display in a table with a toolbar to perform actions.



Standard toolbar actions are covered in the working with Longview tables section in the *Longview Solutions Framework User Guide*.

The toolbar contains the following additional actions:

1. Submit: Click to submit the changes to the process maps.
2. Refresh: Click to refresh the list of process maps. If there are any unsaved changes you will be prompted to save or discard them.

The table includes the following columns:

Column	Notes
ID	The row ID used in validation messages.
Published	Indicates whether the process map has been published. **Note:** The ID should only contain alphanumeric characters. Published process maps cannot be deleted
Process ID	The ID of the process map. This field is editable only for new process maps. **Note:** Process ID's must be unique. You can not have two process maps in one system with the same process ID.
Process Name	The name of the process map that is displayed in the navigation pane of Longview Client.
Process Title	The title of the process map that is displayed when the process map is rendered. **Note:** ▪ You can display the name of a symbol selected from the navigation pane with the following syntax: %%DimensionName%%. ▪ You can display the description of a symbol selected from the navigation pane with the following syntax: <%[[ATTRIBUTE,SYMBOL,DESCRIPTION,%% DimensionName %%]]%>

Edit Process Map

The Edit Process Map app allows you to edit a selected process map in your system.

Accessing edit process maps

1. Select the Design module
2. Select the Process Maps category.
3. Click Edit Process Map. The Edit Process Map selection dialog appears

Note: You can also access the edit process maps app through the Manage Process Maps app by right-clicking on the row containing the process map you would like to edit and selecting the edit option.

Editing Process Maps

To edit a process map:

1. The Select Process Map dialog appears. Select the Process Map you would like to edit.
2. Click OK. The Edit Process Map table appears.

Adding a process step

Click Add to add a new step to your process map. Steps are not added until you click Apply.

Note: You can add a new process map step that has similar settings by selecting an existing row and clicking duplicate.

For each process map step, fill in the following fields:

Field	Description
Step ID	Enter an ID for the process step. **Note:** Step ID can not be left blank. Step ID's are unique. You can not have two steps in the same process map with the same step ID.
Step Name	Enter a name for the process step. **Note:** Step name can not be left blank.
Description	Optionally enter a description for the process step. The description appears as a tooltip on the process map step and can be up to 1,024 characters. If no description is provided the tooltip will display the step name.

Field	Description
Action Type	Select the action type for the process step. This, in conjunction with the Action Link, defines what will be launched clicking on the process map step. The following Action Types are available: ■ App - Will launch a specified app. ■ File - Will launch a specified file. ■ Longview Component - Will launch a specified Longview Component. ■ Process Map - Will launch a specified process map. ■ Report - Will launch a specified report. ■ System App - Will launch a specified system app. ■ Web - Will launch a specified URL. ■ Workflow – Will launch a workflow app that will allow you to change the status of a specified workflow process and step.
Action Link	Select the Action Link that will occur when you click on the process step. The Action Link will depend on the Action Type selected
Icon	Select the icon that will display on your process map step. Depending on the Action Type and Action Link selected, a default icon will be selected for you. This can be changed if required.
Color	Select the color that will display for your process map step. Depending on the Action Type and Action Link selected, a default color will be selected for you. This can be changed if required.
Next Steps	Next steps allow you to create a structured process map, with dependencies between steps. Select the step(s) that will come directly after the step being edited. **Note:** ■ If no next steps are selected within a process map, the process map will remain unstructured. ■ If a recursive cycle is created when setting up a process map, the process map may not render when being displayed.
Excel File Name	Select an excel file to launch if action type is Longview Component and Action Link is Longview Add-In for Office. **Note:** This field is optional. If no file is defined, the Longview Add-In for office will launch without opening a specific file.

Action Link Options

The following action link options are available, depending on the action type selected:

Action Type	Action Link
App	A list of apps that are published in your system are available for selection.
File	A file browser that allows you to select the file to launch when clicking on the process step is available.
Longview Component	The following components can be selected: ■ Longview Add-In for Office ■ Longview Application Administrator ■ Longview Analysis and Reporting ■ Longview Dashboard Designer ■ Longview Journal Entries ■ Longview Server Manager ■ Longview Workflow Designer
Process Map	A list of published process maps (except for the current process map being edited) are available for selected.
Report	A list of reports that are published in your system are available for selection.

Action Type	Action Link
System App	The following system apps can be selected: Global system apps (will be available in all systems) ■ Apps Publisher ■ Data Locks ■ Events ■ Events Status Tax system apps: **Note:** These system apps will only be available in a Longview Tax system. Not all apps may be available, this depends on the configuration of your Longview Tax application (for example, if your tax application contains multi-regional provision or task management, additional system apps will be available). ■ Regions ■ Rollover ■ Rollover and Scenario Setup ■ System Settings ■ Tax Account Rollforward ■ Tax Account Rollforward Data Transfer ■ Tax Calculations ■ Book Tax Differences ■ Currencies ■ Current and Deferred Tax ■ Entities ■ Export ■ Import ■ Interim ■ Jurisdictions

Action Type	Action Link
	■ Loss, Tax Credit, and Other Accounts ■ Net Income Before Tax ■ Regional Modifications
Web	Enter a URL that will be launched when clicking on the process map step.
Workflow	Select a workflow process and associated workflow step. When the step is launched, the user will be able to change the status of the associated process step, depending on their level of workflow access (reviewer, owner, or approver).

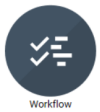
Default icons and colors

The following icons and colors are defaulted depending on the action type and action link selected:

Action Type	Action Link	Default Icon	Default Color	Preview
App	Selected App	data_collection	#1173CB	
File	Any File	folder_file	#7A8791	
Longview Component	Longview Add-In for Office	export_excel	#1AAC35	
Longview Component	Longview Application Administrator	cog	#EA3B93	
Longview Component	Longview Analysis and Reporting	note	#EE7B23	
Longview Component	Longview Dashboard Designer	edit	#A91C3C	
Longview Component	Longview Journal Entries	journal_entries	#2A588A	
Longview Component	Longview Server Manager	server_ manager	#696A6C	

Action Type	Action Link	Default Icon	Default Color	Preview
Longview Component	Longview Workflow Designer	itemize	#696A6C	
Process Map	Selected Process Map	process_maps_1	#FEC53E	
Report	Selected Report	note	#EE7B23	
System App	■ Data Locks ■ Events ■ Event Status ■ User Submissions	monitoring	#902C8B	
System App	■ Users ■ User Groups	collaborate	#21CCC3	

Action Type	Action Link	Default Icon	Default Color	Preview
System App	■ Tax Calculations ■ Jurisdictions ■ Loss, Tax Credit, ■ and Other ■ Accounts ■ Net Income ■ Before Tax ■ Regions ■ Regional ■ Modifications ■ Rollover ■ Rollover and ■ Scenario Setup ■ System Settings ■ Tax Account ■ Rollforward ■ Tax Account ■ Rollforward Data ■ Transfer ■ Apps Publisher ■ Book Tax ■ Differences ■ Current and	Thumbs	#EA3B93	

Action Type	Action Link	Default Icon	Default Color	Preview
	▪ Deferred Tax ▪ Currencies ▪ Entities ▪ Export ▪ Import ▪ Interim			
Web	Entered URL	global	#6ABB5E	
Workflow	Selected workflow process and workflow step	itemize	#425363	

Deleting a process step

Delete a process step by selecting the row and clicking delete. The process step is not deleted until you click Apply.

Note: Deleting a process step will automatically delete it from any next steps where it was listed.

Editing a process step

You can change the properties of a process step by selecting different values in edit process map table. No changes are applied until you click Apply.

Applying changes

To apply changes to the process map steps, click Apply.

Note: Step ID's and Step Names cannot be left blank.

Refreshing the process maps

Click Refresh to refresh the list of process maps.

Import Process Maps

The import process maps app allows an administrator to import one or more process maps from the a .json file. This file can be used to copy the selected process maps to another system. Import process

maps is found in the Process Maps category of the Design module.

To access import process maps:

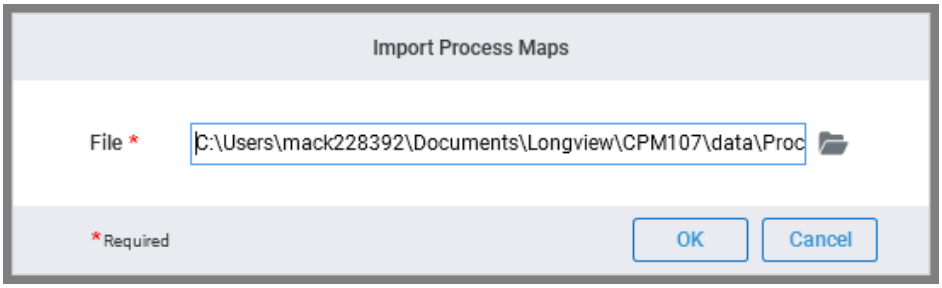
1. Select the Design module
2. Select the Process Maps category.
3. Click Import Process Maps.

Importing process maps

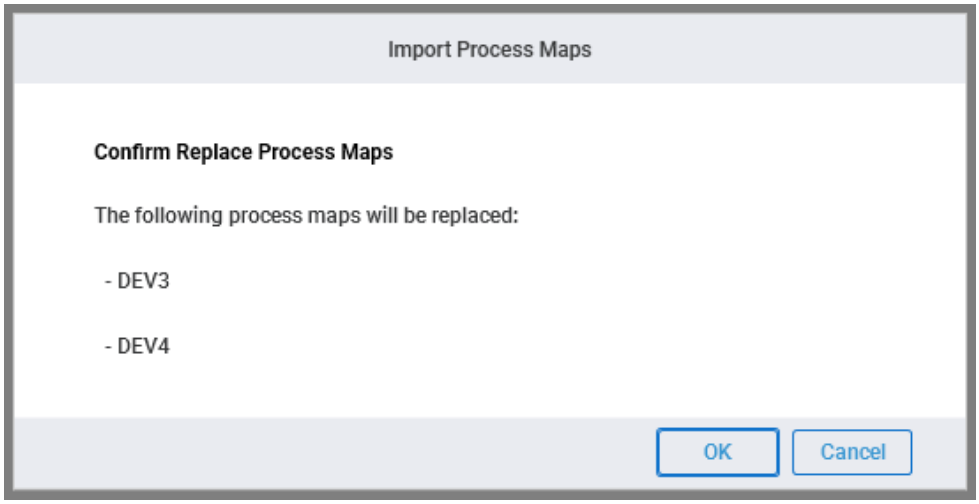
The import process maps process allows you to import new and/or replace existing process maps.

To import process maps:

1. Select the file to import.
   a. The export process maps app stores exported process maps files in `Documents\Longview\<LVID>\data\Process Maps`.



2. Click OK.
3. If any process maps in the selected file will replace process maps already in the system a confirmation dialog will appear.

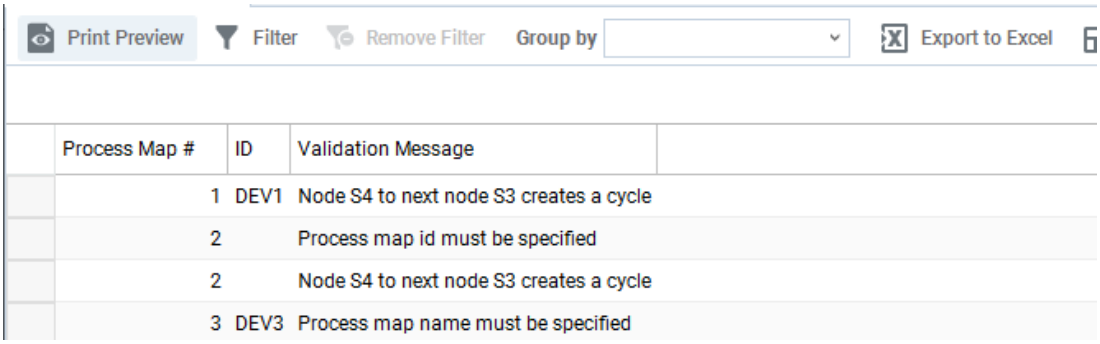


4. Click OK
5. The process map file will be validated.

6. If there are any validation errors, a table listing the errors will appear. No process maps are imported.
7. Otherwise, a process complete message will appear.

Reviewing validation errors

In normal cases validation errors occur only if the file imported was manually modified, which is not recommended. If there are validation errors the following table is displayed:



Column	Notes
Process Map #	This indicates the process map in the source file that the validation error is related to. It is provided specifically to help the process map when the id is not specified.
ID	This is the ID of the process map (“id” in the source file).
Validation Message	The issue detected by the validation. If the issue is related to a node property, the node number and id are included in the message. If the issue is related to a cycle the node and next node that cause the cycle are identified by the node id.

Export Process Maps

The export process maps app allows an administrator to export one or more process maps from the system to a .json file. This file can be used to copy the selected process maps to another system. Export process maps is found in the Import and Export folder of the Administration category.

Accessing export process maps

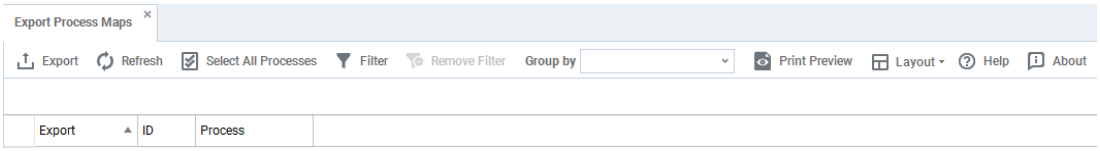
1. Select the Design module
2. Select the Process Maps category.
3. Click Export Process Maps.

Exporting process maps

When the export process maps app is selected the user is shown a datatable listing the process maps in the system.

Select the one or more of the process maps to include in the file and click Export. Once the export has been run selected process map definitions are downloaded to the `\Documents\Longview\<LID>\data\Process` Maps folder with a file name of `Process Maps_<TimePeriod>_YYYMMDD_HHMMSS.json`

The export process maps app display includes a table with a toolbar to perform actions.



Standard toolbar actions are covered in the working with Longview tables section in the *Longview Solutions Framework User Guide*.

The toolbar contains the following additional actions:

1. Export: Click to export the selected process map definitions to a .json file.
2. Select All Processes: Click to select all available processes for export.

The table includes the following columns:

Column	Notes
Export	Check boxes allowing the user to select which process maps to include in the export.
ID	The unique IDs of the individual process maps in the system.
Process	The process map name as seen in the navigation pane of the system.

APIs Category

The apps category contains items based on their purpose.

Administrative toolbar

Longview Designer includes an administrative toolbar. The buttons that appear on the administrative toolbar may vary depending on the tool or editor you have open in the workspace:

Button	Description
Save	Click the Save button to save any changes made in the current view.
Duplicate	Click the Duplicate button to create a new item or app with the settings in the current view.
Help	Click the Help button to open context-sensitive Help related to the appropriate tool or editor.

Specifying properties

Properties are general information about the API. The name of an API is only referenced in the REST URI and generally should be kept short.

Field	Notes
Name	Type a name for the API. The name must be unique and can have a maximum of 94 characters, including spaces. You cannot include the slash (/), backslash (\ ), colon (:), ampersand (&), asterisk (*), question mark (?), double quotation mark ("), the right angle bracket (>), left angle bracket (<), pipe (|), or period (.) in the app name.
Description	Type a description for the app. The description can have a maximum of 100 characters, including spaces.
Version	Type a version to display in brackets after the description in the title bar for the data import app. The version can have a maximum of 100 characters, including spaces. This field is optional.
Template version	Type a version to display in brackets after the description in the title bar for the data import app. The version can have a maximum of 100 characters, including spaces. This field is optional.

Create an API

You can create an API by:

- Starting with a template, or
- Duplicating an existing API

To create a new API, event, or configuration library from a template:

1. Navigate to the Design module.
2. Expand the folder that relates to the purpose of the API you which to create.

3. Click New.
4. A new tab appears with the properties page open.

To create a new API by duplicating an existing one:

1. Navigate to the Design module.
2. Expand the folder that relates to the purpose of the API you which to create.
3. Expand the Existing folder under the specific type of API.
4. Right-click on the item you wish to copy and select Duplicate.
5. A new tab appears with the properties page open, containing all the settings and documents in the duplicated item.

Delete an API

If you have the appropriate authorization, you can use Longview Client to delete an API.

Creating data get APIs

You can use Longview Designer to create APIs that will export data from the Longview database. The target file may contain a single column of values or multiple columns of values. Depending on the type of export you would like to create, you can use either the Data GET – Single Value or Data GET – Multi value template.

> **Note:** Variables can be used but they will need to be passed as a parameter along with their value in the REST URI or a 500 Internal Server Error will be returned.

Specifying properties

You can use Longview Designer to specify properties for a new data API.

For more information, see Specifying properties.

Specifying the Data Area Definition

Use the Data Area Definition to create a dataspec document (.lvdsp) used to define the area to be exported.

1. In the Options table, complete the following field

Field	Description
Data Type	Specify the data to be queried using one of the following options: ■ ADJUSTED — Indicates that data queries include adjustments made via journal entries ■ UNADJUSTED — Indicates that data queries exclude adjustments made via journal entries Default is ADJUSTED.
Download Type	Specify the type of data to be downloaded using one of the following options (download option specific in brackets): ■ All data — Indicates that all data within the data area definition is downloaded (STANDARDALL). ■ Exclude parent values — Indicates that parent data is excluded in the download (STANDARDLEAFONLY) ■ Exclude parent and calculated values — Indicates that only leaf data that was submitted to the database via import, input or event calculation is included in the download (LEAFDATA). Default is All data.

2. In the Data Area table, complete the following:

- Schedule: This field displays only if your system is configured to use schedules and you have access to schedules.
  - The default of None indicates the export spec is for base data only.
  - Specify the schedule to use for the import spec, and the schedule dimensions appear at the bottom of the list of Dimensions.
- Complete the following fields for each dimension:

Note: The Spec and Level fields are not applicable if a symbol contains a variable.

Field	Description
Symbols	Type a symbol name. Alternatively, you can use the symbol selector. To add more than one symbol for a dimension, select a row containing the dimension for which you want to add another symbol and click Symbol. To move a symbol up or down within a dimension, click Move Up or Move Down.

Field	Description
Spec	Indicate the symbol specification using one of the following values: • All — Includes all symbols within the selected symbol's hierarchy. • Leaf — Includes only leaf symbols within the selected symbol's hierarchy. • Parent — Includes only parent symbols within the selected symbol's hierarchy. • Root and Parent — Includes only root and parent symbols within the selected symbol's hierarchy.
Level	Specify the number of levels down from the symbol to be included in the data area.

3. Use Additional Configuration to specify any other Data Area Definition features you wish to use. Additional Configuration is a code editor that allows you to enter data area definition functions to be included in the data area.
   For more information on the code editor, see Using the code editor.

   Typical use of Additional Configuration in a data area definition would be to:

```
# Add an attribute filter using the AttributeFilter function
```

Specifying the Data Area Export Spec

Note: Exported files will be in CSV format with comma (,) delimiters and period (.) decimals.

To create the data area export spec:

1. For a multiple value, file-based export: in the Values Dimension table complete the following field:

Field	Description
Dimension corresponding to values	Specify the dimension for which there are multiple value fields.

2. The Dimension Definitions section is used to specify how each dimension is processed during export. For Dimension Definitions, complete the following:

   - Schedule: This field displays only if your system is configured to use schedules and you have access to schedules.

     a. The default of None indicates the export spec is for base data only.

     b. Specify the schedule to use for the import spec, and the schedule dimensions appear at the bottom of the list of Dimensions.

- For each dimension, complete the following fields:

Field	Description
Dimension	The name of the dimension.
Type	Specify the type of relationship between dimension symbols and field values in the data source using one of the following values: • Map — The dimension symbols are mapped to values as specified in the indicated map • Match — The dimension symbols are written out with their names.
Field Name	This field is available only when Type is Map or Match. Optionally, enter the name of the field. If the field name is not specified it will be set to Field with the Field Position appended. The field name is used to specify filters. If the target data is in JSON format the field name also specifies the name of the property.
Field Position	This field is available only when Type is Map or Match. Specify the field position to assign the dimension to using a positive integer. For example, for the Accounts dimension, type 3 if the account will be the third column of the data target.
Symbol	This field is available only when Type is Unique. Type the symbol name for the specified dimension to which the data is imported. Alternatively, you can use the symbol selector. **Note:** You can also use a variable to define the symbol.
Map Location	This field is available only when Type is Map. Indicate the location of the map using one of the following values: • Internal — Indicates the map exists inside the database, created using the Mappings editor. For more information, see Maintaining mappings. • External — Indicates the map exists as a file, such as a text file on your local machine or network location. • Document — Indicates the map exists as a document within the Symbol Maps folder of the app.

Field	Description
Map	This field is available only when Type is Map. Do one of the following: • If you selected Internal as the Map Location, select a map from the list of available maps in the system. • If you selected External as the Map Location, type the path of the external map or click the ellipsis button (...) to select the external map. • If you selected Document as the Map Location, select a map from the list of available maps in the app. **Note:** You can use a variable for the Map field if the map location is Internal or External.

3. For a single value, file-based export: in the Value Field table complete the following field:

Field	Description
Value field name	Specify the name of the field that contains the value. The field name is used to specify filters. The default value for this field is 'value'. If the target data is in JSON format the field name also specifies the name of the property.
Value field number	Specify the field position at which the data value will be written using a number.
v26.2 Value field decimals	Specify the number of decimal places for exported numeric values. This setting controls the precision of the exported numbers. If not specified, the default is 9 decimal places.

4. If the data target contains multiple value fields, the Value Fields section is available. Complete the following fields:

Field	Description
Field Name	Optionally, enter the name of the field. If the field name is not specified it will be set to Field with the Field Position appended. The field name is used to specify filters. If the target data is in JSON format the field name also specifies the name of the property.

Field	Description
Field Position	Specify the field position of the value field using a positive integer, where the field position is the corresponding column in the data source containing the data. For example, if the value is exported to the third column in the target file, type 3.
Symbol	Type the name of a symbol from the dimension indicated in Multiple Value Fields for Dimension for the value in the corresponding field position. Alternatively, you can use the symbol selector. **Note:** You can use a variable for the Symbol field.

5. In the Data Options table, complete the following fields:

Field	Description
Reverse value for credit accounts	Select this option to reverse the sign for numeric values for accounts with Credit as the Balance Type. For more information on the balance type in the ACCOUNTS dimension, see "Working with dimensions" in the *Longview Application Administrator Guide*. The default is No.
Allow duplicate mappings	This option applies only when at least one dimension Type is Map. Select this option to indicate that duplicate mappings are permitted. If multiple mappings are found, the system does the following for each type of mapping: ■ Exact — All duplicate Exact mappings are used so that the same value is imported to multiple data cells. If there are duplicate Exact and Wildcard or Range mappings, the Exact mapping is used, and the Wildcard or Range mapping is ignored. ■ Wildcard — If no Exact mappings are found, the value is imported to the first Wildcard or Range mapping listed. ■ Range — If no Exact mappings are found, the value is imported to the first Wildcard or Range mapping listed. ■ If you do not select this option, duplicate mappings are not permitted. If multiple mappings are found, the system reports an error for each mapping. The default is No.

Field	Description
Handling for duplicate records	Specify the way duplicate records are handled for the data import app using one of the following values: ■ Output all — Specifies that each duplicate record will be written to the target file. ■ Error — Specifies that duplicate records should not be allowed. The first entry is submitted. If a duplicate record is encountered, the system reports an error. ■ Sum values — Specifies that duplicate records should be allowed and the value of all such records summed. If a duplicate record is encountered that contains text (versus numeric) data, it is treated as an invalid/error record. The default is Error.
Maximum number of errors	Specify the maximum number of error records to permit before stopping the import process. The default is 0.
Header option	Specify whether to create a header in the target file, using one the following values: ■ None – Specifies that no header will be created and the first record in the file is a data record. ■ Auto — Specifies that a single header record is created in the target file. The header will contain the description of the dimension for each field that is a dimension, and ◦ “Value” for the value field ◦ The description of each symbol in a multiple value export. ■ Custom — Specifies that the header records will be manually entered in the Custom header field.
Custom header	When the header option is set to Custom, enter the header records in the form: “header record”, “...”, “header record”. Each header record is output on a new line in the target file.

6. You can optionally specify filters to apply when exporting data. For each filter type an expression using the following guidelines:

   - The expression can be joined together by two or more expressions using AND or OR and grouped by brackets, as appropriate.
   - Each expression uses field names, operators, and values as follows: FieldName EQ|NE|LE|LT|GE|GT Value. If Value is a string, enclose the string in double quotation marks. • You can optionally include wildcard characters (? and *) and symbols for Value.

     Note: In this case do not enclose Value in double quotation marks.

- You can optionally use symbols for operators, such as ==, !=, <=, <, >=, and >.
- You must use Field# for FieldName to refer to a specific field on which you want to filter. For example, if Products is the fourth column in your target file and you want to filter Products to import only the Products_Default symbol, use Field4 == "Products_Default" as the expression.

> **Note:** For more information on creating expressions, see “Using conditional operators” and “Search” in the *Longview Developer’s Guide*.

Specifying Symbol Maps

Symbol Maps(.lvmap) specify instructions when the symbol names in your data source do not match the symbol names in your Longview database.

For more information, see “Developing Longview ImportSpecs, ExportSpecs, and external Maps” in the *Longview Developer’s Guide*.

For more information on using the code editor, see "Using the code editor".

Creating data put APIs

You can use Longview Designer to create APIs that will submit data to the Longview database. Depending on the format of your source data file, you can use either the Data PUT – Single Value or Data PUT – Multi value template.

> **Note:** Variables can be used but they will need to be passed as a parameter along with their value in the REST URI or a 500 Internal Server Error will be returned.

Specifying properties

You can use Longview Designer to specify properties for a new data API.

For more information, see Specifying properties.

Specifying the Data Area Definition

Use Data Area Definition to create a dataspec document (.lvdsp) used to define the area to be imported to.

1. In the Data Area table, complete the following:
   a. Schedule: This field displays only if your system is configured to use schedules and you have access to schedules.

   i. The default of None indicates the export spec is for base data only.

   ii. Specify the schedule to use for the import spec, and the schedule dimensions appear at the bottom of the list of Dimensions.

b. Complete the following fields for each dimension:

c. The Spec and Level fields are not applicable if a symbol contains a variable.

Field	Description
Symbols	Type a symbol name. Alternatively, you can use the symbol selector. To add more than one symbol for a dimension, select a row containing the dimension for which you want to add another symbol and click Symbol. To move a symbol up or down within a dimension, click Move Up or Move Down.
Spec	Indicate the symbol specification using one of the following values: ■ All — Includes all symbols within the selected symbol's hierarchy. ■ Leaf — Includes only leaf symbols within the selected symbol's hierarchy. ■ Parent — Includes only parent symbols within the selected symbol's hierarchy. ■ Root and Parent — Includes only root and parent symbols within the selected symbol's hierarchy.
Level	Specify the number of levels down from the symbol to be included in the data area.

2. Use Additional Configuration to specify any other Data Area Definition features you wish to use. Additional Configuration is a code editor that allows you to enter data area definition functions to be included in the data area. For more information on the code editor, see Using the code editor.

Typical use of Additional Configuration in a data area definition would be to

Add an attribute filter using the AttributeFilter function.

Specifying the Data Area Import Spec

Data area import specs (.lvimp) are used to import data to a data area. The source data may contain a single column of values or multiple columns of values.

Note: Incoming files are required to be in CSV format with comma (,) delimiters and period (.) decimals. The import file is sourced from the body of the REST request header.

To create a data area import spec:

1. Right-click on the Data Area Import Spec document.
2. Fill in the data source.

Data source	Description
Number of header records to filter	Specify the number of header rows to filter on import. If you leave the value as the default value of 0, the system assumes your data source starts with data records with no header rows.

Data source	Description
Number of footer records to filter	Specify the number of footer rows to filter on import. If you leave the value as the default value of 0, the system assumes your data source ends with data records with no footer rows.

3. For a multiple value, file-based import: in the Values Dimension table, complete the following fields:

Field	Description
Dimension corresponding to values	Specify the dimension for which there are multiple value fields.

4. The Dimension Definitions section is used to specify how each dimension is processed during import. For Dimension Definitions, complete the following:

   - Schedule: This field displays only if your system is configured to use schedules and you have access to schedules.

     a. The default of None indicates the import spec is for base data only.

     b. Specify the schedule to use for the import spec, and the schedule dimensions appear at the bottom of the list of Dimensions.

   - For each dimension, complete the following fields:

Field	Description
Dimension	The name of the dimension.

Field	Description
Type	Specify the type of relationship between dimension symbols and field values in the data source using one of the following values: • Consecutive — The values encountered in the data source are placed consecutively in the dimension starting at the indicated symbol, moving in priority order through the indicated symbol's siblings. **Note:** Consecutive is designed to work with a symbol existing in only one hierarchy. If the symbol belongs to more than one hierarchy, the behavior used to determine the siblings is undefined and may be affected by hierarchy reorganizations and imports/exports. You can specify only one dimension as Consecutive. • Map — The values encountered in the data source are mapped to dimension symbols as specified in the indicated map. • Match — The values encountered in the data source are matched to the dimension symbols exactly. • Unique — All values encountered in the data source are placed in the specified dimension symbol. If you select this option, you can use the variables you defined on the Variables page for the specified symbol.
Field Name	This field is available only when Type is Map or Match. Optionally, enter the name of the field. If the field name is not specified it will be set to Field with the Field Position appended. The field name is used to specify filters. If the source data is in JSON format, the field name must match the name of the property.
Field Position	This field is available only when Type is Map or Match. Specify the field position corresponding to the dimension using a positive integer. For example, for the Accounts dimension, type 3 if the account is in the third column of the data source.
Symbol	This field is available only when Type is Consecutive or Unique. Type the symbol name for the specified dimension to which the data is imported. Alternatively, you can use the symbol selector. **Note:** You can also use a variable to define the symbol.

Field	Description
Map Location	This field is available only when Type is Map. Indicate the location of the map using one of the following values: • Internal — Indicates the map exists inside the database, created using the Mappings editor. For more information, see "Maintaining mappings". • External — Indicates the map exists as a file, such as a text file on your local machine or network location. • Document — Indicates the map exists as a document within the Symbol Maps folder of the app.
Map	This field is available only when Type is Map. Do one of the following: • If you selected Internal as the Map Location, select a map from the list of available maps in the system. • If you selected External as the Map Location, type the path of the external map or click the ellipsis button (...) to select the external map. • If you selected Document as the Map Location, select a map from the list of available maps in the app. **Note:** You can also use a variable for the Map field if the map location is Internal or External.

5. For a single value, file-based import: in the Value Field table complete the following field:

Field	Description
Value field name	Specify the name of the field that contains the value. The field name is used to specify filters. The default value for this field is 'value'. If the source data is in JSON format, the field name must match the name of the property.
Value field number	Specify the field position at which the data value will be written using a number.

6. If the data source contains multiple value fields, the Value Fields section is available. Complete the following fields:

Field	Description
Field Name	Optionally, enter the name of the field. If the field name is not specified it will be set to Field with the Field Position appended. The field name is used to specify filters. If the source data is in JSON format, the field name must match the name of the property.
Field Position	Specify the field position of the value field using a positive integer, where the field position is the corresponding column in the data source containing the data. For example, if the value is in the third column in the import file, type 3.
Symbol	Type the name of a symbol from the dimension indicated in Multiple Value Fields for Dimension for the value in the corresponding field position. Alternatively, you can use the symbol selector. **Note:** You can use a variable for the Symbol field.

Note: You can delete a value field by selecting a value field row and clicking Delete.

7. In the Data Options table, complete the following fields:

Field	Description
Data Import Method	This field identifies if the source file will always have a complete replacement of the data in the target dataarea or if it will support partial changes. ■ Full – All data for the target space will be included in the source file, any records not included in the file will be assumed to be zero values. ■ Incremental – Only those records included in the import file will be uploaded to the target space
Reverse value for credit accounts	Select this option to reverse the sign for numeric values for accounts with Credit as the Balance Type. For more information on the balance type in the ACCOUNTS dimension, see "Working with dimensions" in the *Longview Application Administrator Guide*. The default is cleared.

Field	Description
Allow duplicate mappings	This option applies only when at least one dimension Type is Map. Select this option to indicate that duplicate mappings are permitted. If multiple mappings are found, the system does the following for each type of mapping: ■ Exact — All duplicate Exact mappings are used so that the same value is imported to multiple data cells. If there are duplicate Exact and Wildcard or Range mappings, the Exact mapping is used, and the Wildcard or Range mapping is ignored. ■ Wildcard — If no Exact mappings are found, the value is imported to the first Wildcard or Range mapping listed. ■ Range — If no Exact mappings are found, the value is imported to the first Wildcard or Range mapping listed. If you do not select this option, duplicate mappings are not permitted. If multiple mappings are found, the system reports an error for each mapping. The default is No.
Handling for duplicate records	Specify the way duplicate records are handled for the data import app using one of the following values: ■ Use first record — Specifies that the first entry of duplicate records should be used. If a duplicate record is encountered, the system does not report an error. ■ Use last record — Specifies that the last entry of duplicate records should be used. If a duplicate record is encountered, the system does not report an error. ■ Error — Specifies that duplicate records should not be allowed. The first entry is submitted. If a duplicate record is encountered, the system reports an error. ■ Sum values — Specifies that duplicate records should be allowed and the value of all such records summed. If a duplicate record is encountered that contains text (versus numeric) data, it is treated as an invalid/error record. ■ Use first record — Specifies that the first entry of duplicate records should be used. If a duplicate record is encountered, the system does not report an error. The default is Error
Maximum number of errors	Specify the maximum number of error records to permit before stopping the import process. The default is 0.

Field	Description
Submit if fewer than maximum number of errors	Select this if the Number of errors before processing stops is greater than 0 and you want to allow for the import to submit the valid records and report the error records. If set to FALSE then no submission will occur if there are any errors. The default is FALSE.

8. For each filter type an expression use the following guidelines:

   - The expression can be joined together by two or more expressions using AND or OR and grouped by brackets, as appropriate.
   - Each expression uses field names, operators, and values as follows: FieldName EQ|NE|LE|LT|GE|GT Value. If Value is a string, enclose the string in double quotation marks.
   - You can optionally include wildcard characters (? and *) and symbols for Value.

     Note: In this case do not enclose Value in double quotation marks.

   - You can optionally use symbols for operators, such as ==, !=, <=, <, >=, and >.
   - You must use Field# for FieldName to refer to a specific field on which you want to filter. For example, if Products is the fourth column in your import file and you want to filter Products to import only the Products_Default symbol, use Field4 == "Products_Default" as the expression.

Note: For more information on creating expressions, see “Using conditional operators” and “Search” in the *Longview Developer’s Guide*.

Specifying Symbol Maps

Symbol Maps(.lvmap) specify instructions when the symbol names in your data source do not match the symbol names in your Longview database.

For more information, see “Developing Longview ImportSpecs, ExportSpecs, and external Maps” in the *Longview Developer’s Guide*.

For more information on using the code editor, see Using the code editor.

Administration Category

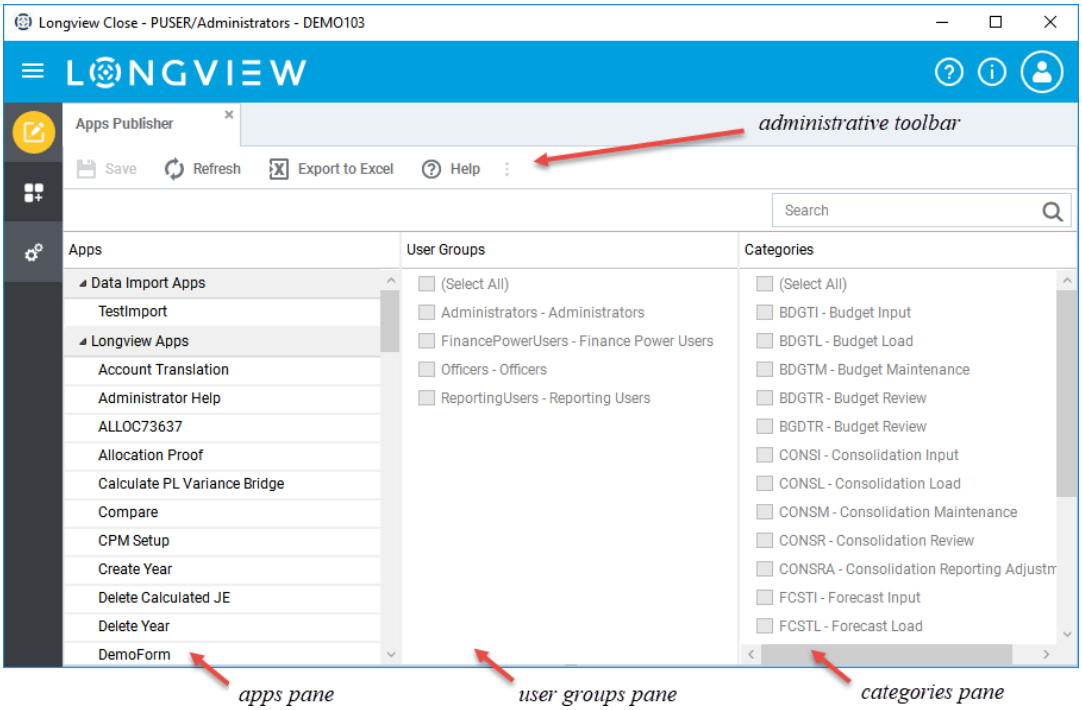
The administration category is divided into folders containing the Apps Publisher, Process Maps Publisher, and the Configure Categories app.

- **Apps Publisher**: Use the apps publisher to assign which groups can access an app, and which categories an app is assigned to. Assigning an app to a category controls which folder it appears in within the Longview Client navigation pane.
- **Process Maps Publisher**: Use the process maps publisher to assign which groups can access a process map, and which categories a process map is assigned to. Assigning a process map to a category controls which folder it appears in within the Longview Client navigation pane.
- **Configure Categories**: Use configure categories to activate and activate standard categories and to configure the folder name of the activated categories. The folder name of the category is used to define the category description and the name of the folder in the navigation pane.

Understanding the Apps Publisher interface

Apps Publisher contains a unique layout to maximize screen space and allow you to view as many apps, user groups, and categories as possible.

Apps Publisher contains an administrative toolbar and is divided into the apps pane, the user groups pane, and the categories pane.



- Administrative toolbar: Apps Publisher includes the Longview Designer administrative toolbar. For more information on buttons on the administrative toolbar, see “Toolbars”.

- Apps pane: The apps pane contains a list of all apps in the system, grouped by app type such as data import apps and Longview Apps. The Longview Apps group contains all Longview Smart Client apps.
- User groups pane: The user groups pane contains a list of all user groups in the system.
- Categories pane: The categories pane contains a list of all categories in the system.

Publishing apps to user groups

Before your app is available for users, you must specify the user group to which the app is available using Apps Publisher. All apps must be published to a user group before users can access them from either Longview Dashboard or Longview Tax.

You can use Apps Publisher to publish all types of apps such as data import apps and Longview Apps.

To publish an app to a user group:

1. Open Longview Designer using the appropriate method.
2. In the Designer navigation pane, click Administration.
3. Click Apps Publisher. Apps Publisher opens in the workspace.
4. In the apps pane, select the app you want to publish to user groups.

   Note: App types for which you do not have the appropriate publishing authorization are not available for modification. Contact your system administrator for more information.

5. In the user groups pane, select the user groups to which you want to publish the app. You must select at least one user group for the app to be available to users. To publish the app to all user groups, use Select All.
6. Click Save. The app is published to the selected user groups.
7. Proceed to Assigning apps to categories.

Assigning apps to categories

You can use Apps Publisher to assign apps to categories. When you assign an app to a category, the app appears listed under the specified category in either Longview Dashboard or Longview Tax so that users can easily view and access apps.

Note: Before you can assign an app to a category, you must first publish the app to at least one user group.

For data import apps, Longview Tax has two preconfigured data import app categories:

Data Imports and Data Imports with Entity Selection. You must assign your data import app to one of these categories so your users can run the data import app.

Note: For data import apps assigned to the Data Imports with Entity Selection category, Longview recommends using the external variable $ENTITIES$ within the data import app.

To assign an app to a category:

1. Open Longview Designer using the appropriate method.
2. In the Designer navigation pane, click Administration.
3. Click Apps Publisher. Apps Publisher opens in the workspace.
4. In the apps pane, select the app you want to assign to categories.
5. In the categories pane, select the category or categories to which you want to assign the app. To assign the app to all categories, use Select All.
6. Click Save. The app is assigned to the selected category or categories.
7. If users access apps from Longview Dashboard, proceed to Displaying apps.

Displaying apps

If users access apps from Longview Dashboard, you need to create a Dashboard link for app so your users can open or run the app.

To create a Dashboard link:

1. In Longview Component Manager, click Longview Dashboard Designer.
2. Open the appropriate Web panel, or create a new one as described in the *Longview Dashboard Designer Guide*.
3. Add the following code to your panel:

```
<a href="Longview.ApplicationSmartClient.application?LongviewWebSID=<%
[[WEBVARIABLE,LongviewWebSID]]%>&LongviewWebBridge=<%
[[WEBBRIDGE]]%>&LongviewIdentifier=<%
[[WEBVARIABLE,LongviewIdentifier]]%>&LongviewApp=AppName&VariableName=Val
ue">Link Text</a>
```

where:

- AppName is the name of the app without the .lvapp extension.
- VariableName is optional and is the variable to which to pass the value. You can pass more than one parameter to the applications as long as the variable is unique.
- Value is the value to pass to the related VariableName.

Note: Values are passed as strings and must use URL encoding if the value contains special characters such as spaces or pipes (|). For example, type Canada%7CUSA for Canada|USA.

- Link Text is the text that you want to appear as the link.

  Example:

```
<a
href="Longview.ApplicationSmartClient.application?LongviewWebSID=<%
[[WEBVARIABLE,LongviewWebSID]]%>&LongviewWebBridge=<%
[[WEBBRIDGE]]%>&LongviewIdentifier=<%
[[WEBVARIABLE,LongviewIdentifier]]%>&LongviewApp=SalaryPlanning&Entit
ies=Toron to"> Toronto Salary Planning </a>
```

  Example with special characters:

```
<a
href="Longview.ApplicationSmartClient.application?LongviewWebSID=<%
[[WEBVARIABLE,LongviewWebSID]]%>&LongviewWebBridge=<%
[[WEBBRIDGE]]%>&LongviewIdentifier=<%
[[WEBVARIABLE,LongviewIdentifier]]%>&LongviewApp=SalaryPlanning&Entit
ies=Canad a%7CUSA"> Toronto Salary Planning </a>
```

4. Save the panel and assign it to a Dashboard page as described in the *Longview Dashboard Designer Guide*.
5. Save the page.

Unpublishing apps from user groups

You can use Apps Publisher to unpublish apps from user groups.

Note: If you unpublish an app from all user groups, all category assignments are removed and the app is no longer accessible to users.

To unpublish an app from a group:

1. Open Longview Designer using the appropriate method.
2. In the Designer navigation pane, click Administration.
3. Click Apps Publisher. Apps Publisher opens in the workspace.
4. In the apps pane, select the app you want to unpublish from user groups.
5. In the user groups pane, clear the user groups from which you want the app unpublished.
6. Click Save. The app is unpublished from the indicated user groups.

Removing apps from categories

You can use Apps Publisher to remove apps from categories.

To remove an app from a category:

1. Open Longview Designer using the appropriate method.
2. In the Designer navigation pane, click Administration.
3. Click App Publisher. Apps Publisher opens in the workspace.
4. In the apps pane, select the app you want to remove from categories.
5. In the categories pane, clear the category or categories from which you want to remove the app.
6. Click Save. The app is removed from the indicated category or categories.

Exporting app access to Microsoft Excel

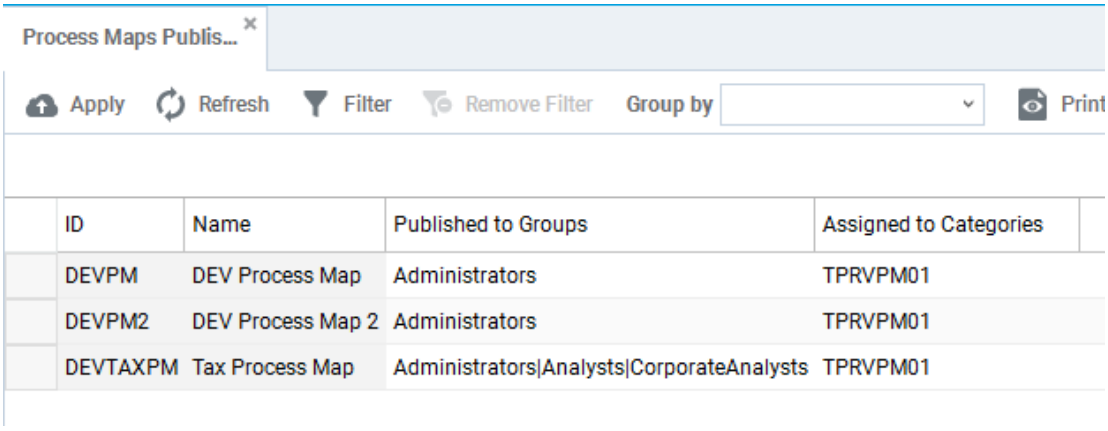
If you have the appropriate authorization, you can use Apps Publisher to export to Microsoft Excel the groups to which apps are published and the categories to which apps are assigned.

To export app access to Microsoft Excel:

1. Open Apps Publisher using the appropriate method.
2. Click Export to Excel. A dialog opens.
3. Specify a name for the export file and click Save. App access is exported to the specified file.

Understanding the Process Maps Publisher Interface

After you have created your process maps, you can make them available for users to run from the Longview Client. Users with the appropriate permissions can use the Process Maps Publisher app to publish process maps to user groups and assign process maps to categories.



Standard toolbar actions are covered in the working with Longview tables section in the *Longview Solutions Framework User Guide*.

The table includes the following columns:

Column	Notes
ID	The ID of the process map.

Column	Notes
Name	The name of the process map
Published to Groups	The groups that the process map is published to.
Assigned to Categories	The catgeories that the process map is assigned to.

Publishing process maps to user groups

Before your process map is available for users, you must specify the user group to which the process map will be available using the Process Maps Publisher app. All process maps must be published to a user group before users can access them from the Longview Client.

To publish a process map to a user group:

1. In the Process Maps Publisher table, find the process map you wish to publish.
2. In the 'Published to Groups' column, click in the cell and select the '...' button to open the list of available groups.
3. Select the groups you would like to publish the process map to and click OK. The list of groups selected will display as a pipe delimited list within the cell.

   Note: You can also key in the user group list as a pipe delimited list into the cell directly. Alternatively, you can copy and paste the user group list from another cell in the table.

4. Click Apply. The process map is published to the selected user groups.

Assigning process maps to categories

You can use the Process Maps Publisher app to assign process maps to categories. When you assign a process map to a category, the process map appears listed under the specified category in the Longview Client so that users can easily view and access process maps.

Note: Before you can assign a process map to a category, you must first publish the process map to at least one user group.

To assign a process map to a category:

1. In the Process Maps Publisher table, find the process map you wish to assign to categories.
2. In the 'Assigned to Categories' column, click in the cell and select the '...' button to open the list of available categories

   Note: Only activated process map categories will be available for assignment.

3. Select the categories that you would like to assign the process map to and click OK. The list of categories selected will display as a pipe delimited list within the cell.

   Note: You can also key in the category list as a pipe delimited list into the cell directly. Alternatively, you can copy and paste the category list from another cell in the table.

4. Click Apply. The process map is assigned to the selected categories.

Unpublishing process maps from user groups

You can use the Process Maps Publisher to unpublish a process map from a user group so they can no longer access it from the Longview Client.

To unpublish a process map:

1. In the Process Maps Publisher table, find the process map you wish to unpublish.
2. In the 'Published to Groups' column, click in the cell and select the '...' button to open the list of available groups.
3. Uncheck the groups that you would like to unpublish the process map from and click OK. The list of remaining selected groups will display as a pipe delimited list within the cell.

> **Note:** If all groups have been de-selected, all assigned categories will also be removed as you cannot have a process map assigned to a category without being published to a group.
> If all groups have been de-selected, any process map that includes a step linking to the unpublished process map will be edited to remove the process map link.

4. Click Apply. The process map is unpublished from all deselected user groups.

Removing process maps from categories

You can use Process Maps Publisher to remove process maps from categories.

To remove a process map from a category:

1. In the Process Maps Publisher table, find the process map you wish to remove from a category.
2. In the Assigned to Categories column, click in the cell, and select the '...' button to open the list of available categories.
3. De-select the categories that you would like to remove the process map from and click OK. The list of remaining categories selected will display as a pipe delimited list within the cell.
4. Click Apply. The process map is removed from all deselected categories.

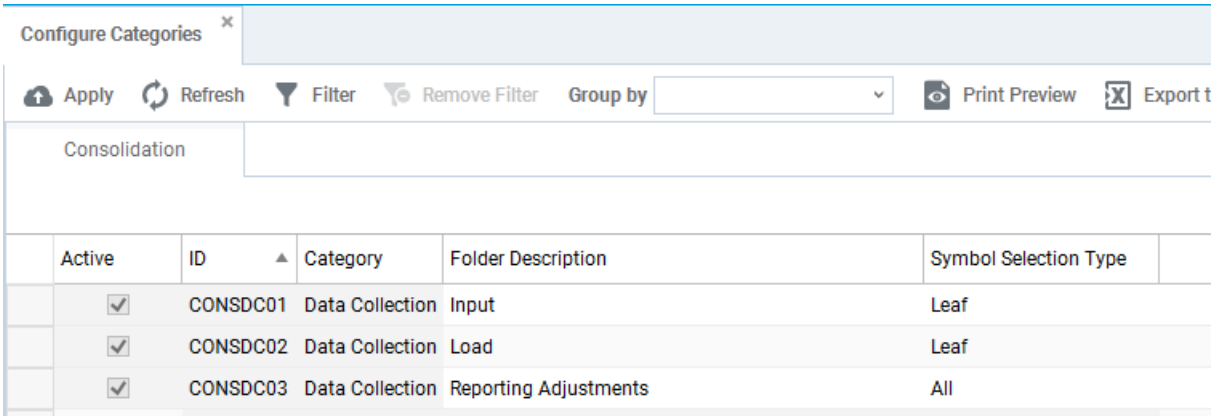
Understanding the Configure Categories Interface

Configure Catgeories allows you to customize the standard folders provide din the Longview Client navigation pane. Depending on your implementation the configure categories table will appear with:

- A combination of the Consolidate tab and the Budget and Forecast tabs.
- A combination of Tax Provision, Task Management, Global Transparency, Tax Planning and Longview Transfer Pricing

Note: Any changes made to existing categories will not be reflected until the navigation pane is refreshed.

Each of the tabs works in the same way, but configures different folders within the Longview client.



Standard toolbar actions are covered in the working with Longview tables section in the *Longview Solutions Framework User Guide*.

The table includes the following columns:

Column	Notes
Active	Indicates whether the publish category is active in the system.
ID	The ID of the publish category.
Category	The category in the client configuration file.
Folder Description	The description of the folder within the navigation. **Note:** ■ The folder displayed for active categories is defined by the folder description. Submit will update the category description and attribute value. ■ The name is limited to 100 characters and cannot contain the following characters: double quotes (“), less than (<), greater than (>) and ampersand (&).
Symbol Selection Type	The symbol selection allowed for the entity selector within the folder. Options include: ■ All: user can select a leaf or parent symbol ■ Leaf: user is only allowed to select a leaf symbol ■ Parent: user is only allowed to select a parent symbol ■ None: no symbol selector will be displayed for the folder

To activate a category:

1. Check the Active column.
2. Enter a description for the folder (mandatory).
3. Select the symbol selection type (mandatory).
4. Click Apply.

The category will be added to the list of publish categories available, and the related system attributes will be updated. The folder will appear in the client navigation once a process map, app or report is published to that category.

To deactivate a category:

1. Open Longview Analysis and Reporting.
2. Click on the Categories tab.
3. Select the category to deactivate.
4. Press the Delete key.

Note: You will only be able to delete a category that has no reports, process maps or apps published to it.

Using The Code Editor

The code editor is used is several templates to allow direct entry of commands, functions and keywords into a document. Depending on the template, a code editor may be used to enter specific functions or an entire document.

Documents

Documents are saved with standard extensions that must be used when referencing the document name in code.

Example: Run a model named Main

```
Run MODEL "Main.lvmod" ON daMain
```

These are the standard extensions.

Document	Extension
Calculated Journal Entry Spec	.lvcje
Data Area Definition	.lvdsp
Data View	.lvdvw
Export Spec	.lvexp
Form	.lvfrm
HTML Document	.html
Import Spec	.lvimp
Model	.lvmod
Model Subroutine	.lvsub
Procedure	.lvpro
Symbol Map	.lvmap
Table Definition	.lvdtd
Table View	.lvtvw
UI	.lvui

Keyboard shortcuts

The code editor supports several keyboard shortcuts.

Movement and selection

The following keyboard shortcuts can be used to navigate through the code editor and select text:

Shortcut	Action
Left, Right, Up, Down, Home, End, Page Up, Page Down	Moves caret

Shortcut	Action
Shift+(Left, Right, Up, Down, Home, End, PageUp, Page Down)	Moves caret with selection
Ctrl+A	Selects all text
Ctrl+Home	Go to first character of the text
Ctrl+End	Go to last character of the text
Shift+Ctrl+Home	Go to first character of the text with selection
Shift+Ctrl+End	Go to last character of the text with selection
Ctrl+Left	Go to next word to the left
Ctrl+Right	Go to next word to the right
Shift+Ctrl+Left	Go to next word to the left with selection
Shift+Ctrl+Right	Go to next word to the right with selection
Ctrl+-	Go back to previous caret location in the editor
Shift+Ctrl+-	go forward to next caret location in the editor

Searching

Shortcut	Action
Ctrl+F	Shows Find dialog
Ctrl+H	Shows Replace dialog
F3	Find next
Ctrl+G	Shows GoTo dialog
Alt+F [character]	Finds nearest [character]

Editing

Shortcut	Action
Ctrl+C	Copy selected text to clipboard
Ctrl+V	Paste text from clipboard
Ctrl+X	Copy selected text to clipboard and delete
Ctrl+Z	Undo operation
Ctrl+R	Redo operation
Tab / Shift+Tab	Increase/decrease left indent of selected range
Ctrl+U/Shift+Ctrl+U	Converts selected text to upper / lower case
Ctrl+Shift+C	Inserts/removes comment prefix in selected lines
Insert	Switches between Insert Mode and Overwrite Mode

Shortcut	Action
Ctrl+Backspace / Ctrl+Del	Remove word left / right
Alt+Up / Alt+Down	Moves selected lines up / down
Shift+Del	Removes current line

Note: To quickly select a line, click on the line number. Select multiple contiguous lines by clicking on a line number and dragging the mouse while holding down the mouse button.

Boomarks and Macros

Note: Bookmarks and macros only persist in the current document editor while it is active.

Shortcut	Action
Ctrl+B	Add a bookmark
Ctrl+Shift+B	Remove a bookmark
Ctrl+N	navigate to next bookmark
Ctrl+Shift+N	Navigate to previous bookmark
Ctrl+M	Start / stop macro recording
Ctrl+E	Execute macro

Others

Shortcut	Action
Esc	Closes all opened tool tips, menus, and hints
Ctrl+Mouse Wheel	Zooming
Ctrl+Up	Scroll up
Ctrl+Down	Scroll down
Ctrl+NumPadPlus	Zoom in
Ctrl+NumPadMinus	Zoom out
Ctrl+0	Reset zoom level to default

Using quick selectors

The code editor provides selectors to make inserting names into your code quicker and easier.

Note: You can select text in the editor before invoking the symbol selector to have the selected symbol replace the highlighted text.

To access the symbol selector:

1. In any code editor, right-click within the code editor area. A pop up menu will appear.
2. Highlight Symbol and select the dimension in which the symbol you require is located.
3. Once the dimension is selected, a symbol selector will appear where you can select your required symbol.
4. Click OK. The symbol name will be entered into the code editor at the spot your cursor is located.

> **Note:** If a numeric symbol name was selected (i.e: 1050), a ! will be prefixed to the symbol name in a model document only. This is required for any numeric symbol in a model where a spec is not defined. If a spec is defined for a numeric symbol (i.e: 1050###), the ! would need to be removed from the syntax. ! are not prefixed to numeric symbols in any other document type (i.e: procedures).

To access the dimension selector:

1. In any code editor, right-click within the code editor area. A pop up menu will appear.
2. Highlight Dimension and select the dimension.
3. The dimension name will be entered into the code editor at the spot your cursor is located.

Using syntax templates

The code editor provides syntax templates in the form of functions, keywords, function templates and in some cases sample documents. Start to type a function and pause typing to be presented with a selection list:

For example, if you type ‘cre’ and pause typing the selection list will present:

- valid functions that start with ‘cre’ are presented first
- any keywords that start with ‘cre’ are presented next
- templates for functions starting with ‘cre’ are presented last

Select an item from the list and press the Tab key or double-click the item to insert it into the editor, or continue typing to narrow the choices.

Understanding syntax conventions

The following conventions are used in the code editor:

- functions are highlighted with bold blue text and are presented in upper camel case.
- keywords are highlighted with bold black text and are presented in uppercase
- identified names (for example dimension names and attribute names) are highlighted with an underline
- in a function template parameters to be manually entered are in default text color and presented in lower camel case

- in a function template parameters that have defined possible values are in default text color and presented in lower camel case followed by an underscore ('_')
- dimensionName is a parameter that can be replaced using the dimension selector. See Using quick selectors.
- symbolName is a parameter that can be rpelaced using the symbol selector. See Using quick selectors.
- #symbolSpec indicates a hierarchical symbol specification. You can invoke a list of symbol specifications by placing the caret within 'symbolSpec' and pressing Ctrl+SPACE.

  Example: Create SYMBOL command template:

  ```
  Create SYMBOL dimensionName symbolName "description" symbolType_
  childSortOrder_ balanceType_
  ```

  - Create is the function
  - SYMBOL is a keyword
  - dimensionName, symbolName and description are parameters to be entered manually
  - symbolType_, childSortType_ and balanceType_ are parameters with defined values

In either case you can replace the parameter by selecting it and typing the desired value. In the case of parameters with defined possible values you can also do one of the following to open a list of possible values:

- place the caret in the word and press Ctrl+SPACE
- place the caret at the end of the word and press CTRL+SPACE

Using document templates

For some document types you can insert a complete document template using the code editor.

Template documents are supported for the following code editor types:

1. Calculated journal entry specifications
2. Data views
3. Export specifications
4. Forms
5. Import specifications
6. Symbol maps
7. Table definitions
8. Table views
9. UI definitions

To insert a template for one of these types do the following:

- start typing the word ‘template’ into the code editor
- If templates are available a list of them will be displayed. In some cases, there is only a single template, while in other cases multiple templates are available.

Adding Context Links To Reports And Apps

Solutions Framework provides out-of-box apps that can be set up as context links within reports and apps.

To make a context link available in a report:

1. Launch Longview Analysis and Reporting.
2. Open a report template.
3. Select Links > Attach Links... The Attach Links dialog appears.
4. For each app check the link to add it to the context menu for the report

App	Cell Context	Template Context
Show Comments	ShowCommentsCell	ShowCommentsTemplate
Show Underlying Adjustments	ShowUnderlyingAdjustments	
Show Audit Trail	ShowAuditTrail	
Show Counterparty Adjustments	ShowCounterPartyAdjustments	

Note: Show Counterparty Adjustments can only be used with the Longview Transfer Pricing package.

To make a context links available in an app:

1. Open an app that contains a data view.
2. Navigate to the Additional Configuration section of a Data Area View (or the code window of a script-based Data Area View).
3. In the code editor type “Action_” and select the appropriate code template from the list:

App	Toolbar
Show Comments	Action_Template_Dataview_Toolbar_ShowComments

App	Cell Context
Show Comments	Action_Template_Dataview_Cell_Context_ShowComments
Show Underlying Adjustments	Action_Template_DataView_Cell_Context_ShowAdjustments
Show Audit Trail	Action_Template_DataView_Cell_Context_Audit
Show Counterparty Adjustments	Action_Template_Dataview_Cell_Context_ ShowCounterPartyAdjustments

Note: Show Counterparty Adjustments can only be used with the Longview Transfer Pricing package.

Using Code Libraries

Solutions Framework provides a set of code libraries that can be used with apps and events. Many of these code libraries are used internally but can also be used when creating configuration libraries and apps to speed development.

Using the ALLOC Code Library

The ALLOC library provides utilities for working with the Manage Allocations App.

For more information on the Manage Allocations App, see the *Solutions Implementer Guide*.

ALLOC global variables

The ALLOC code library creates variables that are used internally, but can also be used within an app. These variables are initialized by procedure ALLOC\Init.lvpro.

Variable Name	Type	Purpose
ALLOC_ <dimension>	RANGE	Stores the available symbols for selection by dimension for: ■ Source ■ Target ■ Pattern ■ Source Offset
ALLOC_ <dimension>Ex	RANGE	Stores the available symbols for selection by dimension for ■ Source Exceptions ■ Target Exceptions **Note:** These variables are set and handled by the ALLOC code library and do not need to be manually set. The ALLOC code library sets these variables depending on the symbols set in the source and/or target data areas.

Setting an allocation area

The ALLOC code library provides a utility for setting a data area in the LVAPP_Areas table. In the Longview Allocation Management app, this is used to set the source, target, pattern, and/or source offset.

This utility will:

- Display a form to select a symbol from each dimension, as defined by the available symbols in the ALLOC_<dimension> variables.

- Write the updated symbol selections to the LVAPP_Areas table. Symbol selections will be stored in the associated AREA_<dimension> variable.

To set an allocation area:

1. Create and download the LVAPP_Areas table

   Note: The datatable must be named dtAreas.

2. Run the ALLOC initialization procedure “ALLOC\Init.lvpro”
3. Set the RELID for the data area entry you want to write to (AREA_RELID)
4. Set the TYPE for the data area entry you want to write to (AREA_Type). For an allocation area, this would be one of the following:

- ALLOCSOURCE
- ALLOCTARGET
- ALLOCPATTERN
- ALLOCSOURCEOF

5. Set the Symbols available for selection for each dimension you want to write into the data area (ALLOC_<dimension>)
6. Run procedure ALLOC\SetArea.lvpro

Example:

```
Create DATATABLE dtAreas USING "Areas.lvdtd"
Download dtAreas
Run PROCEDURE "ALLOC\Init.lvpro"
Set VARIABLE AREA_RELID = 100
Set VARIABLE AREA_Type = "ALLOCSOURCE"
Set VARIABLE ALLOC_ACCOUNTS = "TB#99"
Set VARIABLE ALLOC_TIMEPERIODS = "ACTUAL_YTD#99"
Set VARIABLE ALLOC_ENTITIES = "ENTITIES_Currency#99"
Set VARIABLE ALLOC_DATAVIEWS = "STATUTORY#99"
Set VARIABLE ALLOC_SCENARIOS = "SCENARIOS_ACTUAL#99"
Set VARIABLE ALLOC_CURRENCIES = "CURRENCIES_Source#99"
Run PROCEDURE "ALLOC\SetArea.lvpro"
Upload dtAreas
```

Note: The "ALLOC\SetArea.lvpro" uses the "AREA\Write.lvpro" and "AREA\Retrieve.lvpro" procedures from the AREA code library.
For more information on these procedures, see "Using the AREA code library".

Setting an allocation exception area

The ALLOC code library provides a utility for setting an exception data area in the LVAPP_Areas table. In the Longview Allocation Management app, this is used to set the Source Exception and/or Target Exceptions.

This utility will:

- Display a form to select a symbol from each available dimension.

  Note:
  - Only dimensions where the corresponding source or target data areas contain symbols that are parents will be available for selection in an exception form.
  - If no source or target data areas are set, exceptions cannot be set.
  - If the source or target data areas only contain leaf level symbols, exceptions cannot be set.
  - If the source or target data areas are updated, this may update the source and target exceptions. For example, in the case where a source exception is no longer part of the source data area.

- Write the updated exception symbol selections to the LVAPP_Areas table. Symbol selections will be stored in the associated AREA_<dimension> variable.

To set an allocation exception area:

1. Create and download the LVAPP_Areas table

   Note: The datatable must be named dtAreas.

2. Run the ALLOC initialization procedure "ALLOC\Init.lvpro"
3. Set the RELID for the data area entry you want to write to (AREA_RELID)
4. Set the TYPE for the data area entry you want to write to (AREA_Type). For an allocation exception area, this would be one of the following:
   - ALLOCSOURCEEX
   - ALLOCTARGETEX
5. Run procedure ALLOC\SetExceptionArea.lvpro

Example:

```
Create DATATABLE dtAreas USING "Areas.lvdtd"
Download dtAreas
Run PROCEDURE "ALLOC\Init.lvpro"
Set VARIABLE AREA_RELID = 100
Set VARIABLE AREA_Type = "ALLOCSOURCEEX"
Run PROCEDURE "ALLOC\SetExceptionArea.lvpro"
Upload dtAreas
```

Note: The "`ALLOC\SetExceptionArea.lvpro`" uses the "`AREA\Write.lvpro`" and "`AREA\Retrieve.lvpro`" procedures from the AREA code library.
For more information on these procedures, see "Using the AREA code library"

Using the AREA Code Library

The solutions framework provides a number of code libraries that can be used when developing apps and events. The AREA library provides utilities for retrieving, writing, deleting and duplicating entries in the LVAPP_AREAS table.

LVAPP_AREAS Overview

The LVAPP_AREAS app table is used for storing data area definitions. This app table is used in the Longview Allocations App. When creating this datatable, it must be named dtAreas in order to function with the AREA code library

Example:

```
Create DATATABLE dtAreas USING "Areas.lvdtd"
```

The LVAPP_AREAS app table consists of the following columns:

AREA global variables

The AREA code library creates a number of variables that are used internally, but can also be used within an app. These variables are initialized by procedure AREA_Init.lvpro, which is invoked by all procedures in the AREA library

VariableName	Type	Purpose
AREA_RELID	NUM	Stores the value of the unique identifier for the definition.
AREA_Type	STRING	Stores the value for the type of area definition.
AREA_ TargetID	NUM	Stores the dimension for the data area row

VariableName	Type	Purpose
AREA_<Dimension>	RANGE	Stores the symbol or list of symbols selected for the data area by dimension. **Note:** An AREA_<dimension> variable will be created for each dimension in your system. For example, AREA_ACCOUNTS, AREA_TIMEPERIODS etc.

Writing to LVAPP_AREAS

The AREA code library provides a utility for writing a data area of a specified type to the LVAPP_AREAS table.

To write to the LVAPP_AREAS table:

1. Create and download the LVAPP_Areas table

   Note: The datatable must be named dtAreas.

2. Run the AREA initialization procedure “AREA\Init.lvpro”
3. Set the RELID for the data area entry you want to write to (AREA_RELID)
4. Set the TYPE for the data area entry you want to write to (AREA_Type)
5. Set the Symbols for each dimension you want to write into the data area (AREA_<dimension>)

   Note: AREA_<dimension> is a RANGE variable that can accept a list of symbols for a dimension.

6. Run procedure AREA\Write.lvpro

Example:

```
Create DATATABLE dtAreas USING "Areas.lvdtd"
Download dtAreas
Run PROCEDURE "AREA\Init.lvpro"
Set VARIABLE AREA_RELID = 100
Set VARIABLE AREA_TYPE = "ALLOCSOURCEEX"
Set VARIABLE AREA_ACCOUNTS = "TB"
Set VARIABLE AREA_TIMEPERIODS = "A17_01_YTD"
Set VARIABLE AREA_ENTITIES = "A1|A1SUB"
Set VARIABLE AREA_DATAVIEWS = "GL"
Set VARIABLE AREA_SCENARIOS = "SCENARIOS_A001"
Set VARIABLE AREA_CURRENCIES = "CURRENCIES_Source"
```

```
Run PROCEDURE "AREA\Write.lvpro"
Upload dtAreas
```

Retrieving from LVAPP_AREAS

The AREA code library provides a utility for retrieving a data area of a specified type from the LVAPP_AREAS table into variables.

To retrieve from the LVAPP_AREAS table:

1. Create and download the LVAPP_Areas table

   Note: The datatable must be named dtAreas.

2. Run the AREA initialization procedure “AREA\Init.lvpro”
3. Set the RELID for the data area entry you want to retrieve (AREA_RELID)
4. Set the TYPE for the data area entry you want to retrieve (AREA_Type)
5. Run procedure AREA\Retrieve.lvpro

Example:

```
Create DATATABLE dtAreas USING "Areas.lvdtd"
Download dtAreas
Run PROCEDURE "AREA\Init.lvpro"
Set VARIABLE AREA_RELID = 100
Set VARIABLE AREA_TYPE = "ALLOCSOURCEEX"
Run PROCEDURE "AREA\Retrieve.lvpro"
```

The variables AREA_<dimension> will be created and populated with symbols for each dimension in the specified data area.

Deleting an area from LVAPP_AREAS

The AREA code library provides a utility for deleting a data area of a specified type from the LVAPP_AREAS table.

To delete an area of a specified type from the LVAPP_AREAS table:

1. Create and download the LVAPP_Areas table

   Note: The datatable must be named dtAreas.

2. Run the AREA initialization procedure “AREA\Init.lvpro”
3. Set the RELID for the data area entry you want to delete (AREA_RELID)

4. Set the TYPE for the data area entry you want to delete (AREA_Type)
5. Run procedure AREA\DeleteArea.lvpro

Example:

```
Create DATATABLE dtAreas USING "Areas.lvdtd"
Download dtAreas
Run PROCEDURE "AREA\Init.lvpro"
Set VARIABLE AREA_RELID = 100
Set VARIABLE AREA_TYPE = "ALLOCSOURCEEX"
Run PROCEDURE "AREA\DeleteArea.lvpro"
Upload dtAreas
```

Deleting a set of areas from LVAPP_AREAS

The AREA code library provides a utility for deleting a complete data area set of the same RELID from the LVAPP_AREAS table.

To delete a complete data area from the LVAPP_AREAS table:

1. Create and download the LVAPP_Areas table

   Note: The datatable must be named dtAreas.

2. Run the AREA initialization procedure “AREA\Init.lvpro”
3. Set the RELID for the data area entry you want to delete (AREA_RELID)
4. Run procedure AREA\DeleteRow.lvpro

Example:

```
Create DATATABLE dtAreas USING "Areas.lvdtd"
Download dtAreas
Run PROCEDURE "AREA\Init.lvpro"
Set VARIABLE AREA_RELID = 100
Run PROCEDURE "AREA\DeleteRow.lvpro"
Upload dtAreas
```

Duplicating a set of areas in LVAPP_AREAS

The AREA code library provides a utility for duplicating a complete data area set of the same RELID within the LVAPP_AREAS table.

To duplicate a complete data area within the LVAPP_AREAS table:

1. Create and download the LVAPP_Areas table

   Note: The datatable must be named dtAreas.

2. Run the AREA initialization procedure “AREA\Init.lvpro”
3. Set the RELID for the data area entry you want to duplicate (AREA_RELID)
4. Set the Target ID of the new data area entry being created (AREA_TargetID)
5. Run procedure AREA\Duplicate.lvpro

Example:

```
Create DATATABLE dtAreas USING "Areas.lvdtd"
Download dtAreas
Run PROCEDURE "AREA\Init.lvpro"
Set VARIABLE AREA_RELID = 100
Set VARIABLE AREA_TargetID = 200
Run PROCEDURE "AREA\Duplicate.lvpro"
Upload dtAreas
```

Using the CJE Code Library

The solutions framework provides a number of code libraries that can be used when developing apps and events. The CJE library provides utilities for working with calculated journal entry events.

CJE global variables

The CJE code library creates a number of variables that are used internally but can also be used within a calculated journal entry event. These variables will only be referenced in a calculated journal entry event. These variables are initialized by procedure CJE\Init.lvpro, which is invoked by all calculated journal entry events. Additional information on these variables are provided beneath the table below.

Variable Name	Type	Use
CJE_Type	STRING	Journal entry type
CJE_ ApplicationID	STRING	Application ID
CJE_EntityCode	STRING	Entity code used as part of the application ID when journal entry per entity is created.
CJE_ SubCategory	STRING	The journal entry subcategory to use
CJE_JEDesc	STRING	Journal entry description
CJE_JENotes	STRING	Journal entry notes

Variable Name	Type	Use
CJE_IsFinancial	STRING	Indicates if the journal entry is financial (and thus must balance)
CJE_IsShared	STRING	Indicates if the journal entry is shared
CJE_ IsPostRequired	NUM	Used to determine of the journal entry needs to be created and posted during processing.
CJE_Status	NUM	Used to store the current status of the journal entry

CJE_Type

Used to store the type of the calculated journal entry. This variable is normally set in the Main Procedure of the calculated journal entry event. For more information, see Specifying the main procedure

Values for CJE_Type can be one of the following:

- CURRENTPERIOD: To create a calculated journal entry that adjusts balances in the current period.
- PPA: To create a calculated journal entry that adjusts balances in a prior period.
- RESTATEMENT: To create a calculated journal entry that restates balances in the current period or a prior period. Restatement journal entries are used for changes in accounting rules.
- FUTUREPERIOD: To create a calculated journal entry that adjusts balances in a future period.

Note: Only CURRENTPERIOD and PPA CJE Types are configured to be handled by the calculated journal entry event. The use of RESTATEMENT or FUTUREPERIOD CJE Types would require additional configuration to the calculated journal event.

CJE_ApplicationId

The application ID is the combination of:

- the application ID prefix specified in the calculated journal entry event settings. For more information, see Specifying calculated journal entry settings
- the entity name (see “CJE_EntityCode”)
- two digits for the creation year indicator
- two digits for the creation month indicator.

Note: The complete application ID name can have a maximum of 31 characters. If the entity name is too long and would cause the application ID to be over 31 characters, the entity name will be truncated in the application ID.

CJE_EntityCode

Used to store the entity name to be included in the calculated journal entry application ID. This variable is only used if Create a calculated journal entry by entity is checked in the calculated journal entry event settings.

For more information, see Specifying calculated journal entry settings.

This variable is normally set in the Main Procedure of the calculated journal entry event.

For more information, see Specifying the main procedure

> **Note:** The complete application ID name can have a maximum of 31 characters. If the entity name is too long and would cause the application ID to be over 31 characters, the entity name will be truncated in the application ID.

CJE_SubCategory

Used to store the subcategory to assign the calculated journal entry to. This is specified in the calculated journal entry event settings.

For more information, see Specifying calculated journal entry settings

If you do not specify a value for this parameter, the calculated journal entry subcategory is set to Adjustment.

For more information on subcategories, see the *Longview Application Administrator Guide*.

CJE_JEDesc

Used to store the calculated journal entry description. This is specified in the calculated journal entry event settings.

For more information, see Specifying calculated journal entry settings.

CJE_JENotes

Used to store any detailed notes for the Calculated Journal Entry. This is an optional parameter. The notes field can have a maximum of 1280 characters, including spaces. This is specified in the calculated journal entry event settings.

For more information, see Specifying calculated journal entry settings.

CJE_IsFinancial

Used to Specify if the journal entry is financial. This is specified in the calculated journal entry event settings.

For more information, see Specifying calculated journal entry settings.

CJE_IsShared

Used to Specify if the journal entry is shared. This is specified in the calculated journal entry event settings.

For more information, see Specifying calculated journal entry settings.

CJE_IsPostRequired

Used to determine the conditions under which a calculated journal entry should be posted (ie: When the debit/credit value is not zero). Variable CJE_IsPostRequired should be set to 1 when posting is required and set to 0 when posting is not required. This is normally done within the SetValues procedure.

For more information, see Specifying the set values procedure.

CJE_Status

Used to store the current status of the journal entry. Valid return values are:

- 0 - Does not exist
- 1 - No Status
- 2 – Posted
- 3 – Validated
- 4 – Errors
- 5 - Review

Using the CORE Code Library

The solutions framework provides a number of code libraries that can be used when developing apps and events. The CORE library provides general utilities for use with apps or events.

CORE global variables

The CORE code library creates and sets a number of variables that are used internally, but can also be used within an app. In most cases these variables will only be directly referenced in a free form app or configuration library. These variables are initialized by procedure CORE\Init.lvpro, which is invoked by all Apps and Events.

Variable[.Property] Name	Type	Initial Value
CORE	OBJECT	
CORE.DimensionInfo	OBJECT	
CORE.DimensionInfo. DimensionList	STRING []	List of the names of the base dimensions in the system.
CORE.DimensionInfo. KeyDimensonList	STRING []	This list of the names of the key dimensions: CLOSE: ACCOUNTS,TIMEPERIODS, ENTITIES, CURRENCIES,DATAVIEWS, SCENARIOS TAX: ACCOUNTS,TIMEPER, ENTITIES,CURRENCY, ELEMENTS,CONTROLS
CORE.DimensionInfo. CustomDimensionList	STRING []	The list of the names of the non-key dimensions.
CORE.DimensionInfo. NumCustomDimensions	NUM	The number of custom dimensions.
CORE_DataAreaName	STRING	

Variable[.Property] Name	Type	Initial Value
CORE_DateTime	STRING	yyyyMMdd_hhmmss at the time of execution of the app or event
CORE_Dimension	STRING	
CORE_DimensionIndex	NUM	0
CORE_DimensionList	STRING []	DEPRECTAED: The names of the base dimensions in the system
CORE_ErrorFileName	STRING	$CORE_LogPath$\Error.txt
CORE_FileName	STRING	
CORE_IsNoError	NUM	1 (no error has occurred)
CORE_LogPath	STRING	The path to write log and error files. Path is dependent on: ■ Whether an App or Event is executing ■ If an App is executing, whether Troubleshooting is enabled with the Re-use file for log and history option
CORE_Message	STRING	
CORE_Resource	OBJECT	
CORE_Resource .Name	STRING	
CORE_ Resource.Source	STRING	
CORE_Resource.Args	STRING	
CORE_ Resource.Message	STRING	
CORE_VariableName	STRING	CORE_Message
CORE_Version	STRING	The version and build number of the solutions framework
DEBUG_ ExportDataAreaList	STRING []	
DEBUG_Message	STRING	
LOG_VariableList	STRING []	

Using the get dimension index utility

The get dimension index utility allows you to determine the index of a specific dimension in the database. The index number is stored in the CORE_DimensionIndex variable and is 0-based to align with naming conventions of attributes in the database.

To use the get dimension index utility:

1. Write the following code in a procedure document:

```
Set VARIABLE CORE_Dimension = "<dimensionName>"
Run PROCEDURE CORE\GetDimensionIndex.lvpro
```

2. The index will be returned in the CORE_DimensionIndex variable
3. If the dimension name is invalid the CORE_DimensionIndex variable will be set to the value -1

Using the get resource string utility

The CORE code library contains a utility to retrieve a resource string from a resource file (.lvres). This utility is useful for retrieving a message in the user's language and you are unable to simply use the resource token [[RESOURCE,Source,Name]].

To use the get resource string utility:

1. Set the properties of the CORE_Resource object:

```
Set PROPERTY CORE_Resource.Name = "Message1"
Set PROPERTY CORE_Resource.Source = "Messages"
Set PROPERTY CORE_Resource.Args = "Argument1|Argument2"
Run PROCEDURE "CORE\GetResourceString.lvpro"
```

> **Note:** CORE_Resource.Args is optional and is used to pass values into the message.

2. The message will be retrieved from the resource file and stored in the CORE_Rresource.Message property.

> **Note:** Use of resource args:Resource args are used to replace tokens in the message with actual values. Tokens are specified in the resource using {n} where n is the index of the argument.

Using the example above:

The resource file (Messages.lvres) would contain:

```
Message1 = My message {1} is about {2}.
When get resource string is executed, the value in CORE_Resource. Message
will be set to "My message Argument1 is about Argument2."
```

Using the on close utility

The CORE code library contains an on close utility that is useful in cases where no user input is expected, i.e. for a read-only data table or data area view. The on close utility allows the user to close an App using the familiar X in the top right corner of a window or the X in a tab.

> **Note:** For cases where user input is expected the utility CORE\OnClose.lvpro should not be used.

To use the on close utility:

1. Before the Show command write the following code:

```
OnClose "CORE\OnClose.lvpro"
```

Using the pattern retrieval utility

If patterns ate being used your app, either standalone or in conjunction with the pattern spread tool, you can use the pattern retrieval utility to assist you. The procedure CORE\CreatePatternsLVDSP.lvpro is used to create a data spec named Patterns that can be used to create a data area specification to retrieve patterns. The main requirement for using this utility is to create a RANGE variable named patternTimePeriodList that will hold the time periods to retrieve patterns for.

In additional the utility procedure VIEW\SetPatternPeriods.lvpro can be used to populate this variable for an existing data area.

The full set of template code for pattern retrieval can be inserted using template Patterns_Retrieve to insert the required procedure code.

To insert the template pattern retrieval code:

1. Click in the code editor of a procedure document.
2. Type TEMPLATE_P.
3. Select TEMPLATE_Patterns_Retrieve from the list and press Enter.
4. The following code is inserted:

   TEMPLATE_Patterns_Retrieve

```
//Download Patterns
If SymbolExists(DATABASE, [[SYSTEM,SGPAccountsDimension,DBDEFAULT]],
"[[SYSTEM,SGPAccountsDimension,DBDEFAULT]]" + "_Patterns")
  //Create dataspec for pattern periods
  Create VARIABLE patternPeriodList AS RANGE

  //Determine pattern periods from data area
  Set VARIABLE CORE_DataAreaName = "daMain"
  Run PROCEDURE "VIEW\SetPatternPeriods.lvpro"

  //Create and download data area
  Run PROCEDURE "CORE\CreatePatternsLVDSP.lvpro"
  Create DATAAREA daPattern USING "Patterns"
  Download daPattern
  Unload "Patterns"
```

```
END If
```

Using the rollup model

Depending on the nature of an App or Event it may be required at some point to perform a full rollup of a data area. The CORE code library contains a generic model to perform this operation: Rollup.lvmod.

To execute rollup on a data area:

1. Write the following code

```
Run MODEL "CORE\Rollup.lvmod" ON <DataAreaName>
```

Using the view file utility

During the execution of an App you may have the requirement to display a file to the user. The view file utility opens the specific file is the default program configured in Windows.

For example a ".txt" file may open in Notepad, while a ".csv" file might open in Excel.

To open a specified file:

1. Write the following code:

```
Set VARIABLE CORE_FileName = "<FileName>"
Run PROCEDURE "CORE\ViewFile.lvpro"
```

2. The specified file is opened in the user's default program for the file type

Using the DATA Code Library

The solutions framework provides a number of code libraries that can be used when developing apps and events. The DATA library provides data related utilities for working with app or event.

Converting a data spec document to a data report input file

There may be times when a data report is a convenient and efficient way to extract data from the database. Configuring a data report input file can be complicated. The DATA code library contains a utility to convert a data spec document to a data report input file.

To convert a data spec document to a data report input file:

1. Create a data report header file, specifying the data to output

```
PARENT #data to output: LEAF, PARENT, LEAF_AND_PARENT, VALIDATION, CTA
BOTH #data types to output: UNADJUSTING, BOTH
```

2. Create a data report footer file, specifying the output settings:

```
Y #Expand time periods as columns in output: Y, N
4 #Number of decimals in output
, #field separator in output
"[[VARIABLE,outputFileName]]" #output file
```

> **Note:** For further information on data report input files, see “Creating the ASCII input file for basic database reports” In the *Longview Developer’s Guide*.

3. Create a data spec

4. Write the following code in a procedure document:

```
//Parse data spec into a datareport
Set VARIABLE CORE_FileName = "DATA.lvdsp"
Run PROCEDURE "DATA\DspToDR\ParseDsp.lvpro"
```

> **Note:** This will generate a document named DataReportBody containing the dimension information in the specified data spec document

5. Write the following code in a procedure to complete and execute the data report:

```
Create VARIABLE dataReportFileName AS STRING
Create VARIABLE outputFileName AS STRING
//Generate and execite data report
Set VARIABLE dataReportFileName = "$CORE_LogPath$\DataReport.txt"
Set VARIABLE outputFileName = "$CORE_LogPath$\Output.txt"
Create DOCUMENT "$dataReportFileName$"
Write "$dataReportFileName$" FILE "DataReportH"
Write "$dataReportFileName$" FILE "DataReportBody" APPEND
Resolve "DataReportF" "$dataReportFileName$" APPEND
Save "$dataReportFileName$"
Unload "$dataReportFileName$"
Unload "DataReportBody"
DataReport "$dataReportFileName$" "$outputFileName$"
```

Using the create trial balance utility to fill in custom dimensions

The DATA code library contains a utility to fill in the custom dimensions for a trial balance query dynamically based on system settings.

To create a document containing the custom dimensions:

1. Write the following code in a procedure document:

```
Run PROCEDURE "DATA\CreateTBLVDSP.lvpro"
```

> **Note:** The procedure will create documents in memory containing specifications for the custom dimensions, as follows

Document	Attribute	Hierarchical Specification
TBSrc	ASCFD<?>Source	#0
TBTrg	ASCFD<?>Target	#99
TBSrcDetail	ASCFD<?>Source	#99
TBSrcLeaf	ASCFD<?>Source	###

2. Use the Write command to create a data spec document for all dimensions by appending one of the above documents to a data spec containing the standard dimensions.

Using the EXP Code Library

The solutions framework provides a number of code libraries that can be used when developing apps and events. The EXP library provides utilities for working with export specs in an app or event.

EXP global variables

The EXP code library creates and sets a number of variables that are used internally, but can also be used within an app. In most cases these variables will only be directly referenced in a free form app or configuration library. These variables are initialized by procedure EXP\Init.lvpro, which would need to be called within an app in order to use the variables.

Variable[.Property] Name	Type	Initial Value
EXP	OBJECT	
EXPDelimiter	STRING	,
EXP.ErrorFileName	STRING	$CORE_LogPath$\Export_Error.txt
EXP.FileName	STRING	0
EXP.IsValid	NUM	0
EXP.LogFileName	STRING	$CORE_LogPath$\Export_Log.txt
EXP.MaxErrors	NUM	0

Using the export handling procedure

The export handling procedure provides standard error handling and validation for export processes. The export handling procedure performs the following steps:

1. Initializes Exp.IsValid to 1
2. Executes procedure Export.lvpro (app or event specific procedure)
3. Performs error handling as follows:
   a. If non-export related error (like a syntax error, or missing file), reports error. EXP.IsValid = 0
   b. If export related error, where MaxErrors setting is exceeded: reports error, and allows user to access import log and error file. EXP.IsValid = 0.

i. The log file will be parsed into an object property name EXP.Status, and a summary message provided in string variable EXP_SummaryMessage.

   c. If no error, but export errors less than or equal to MaxErrors setting, provides a message, allowing the user to review errors. EXP.IsValid = 0.

i. The log file will be parsed into an object property named EXP.Status, and a summary message provided in string variable EXP_SummaryMessage.

To use the export handling procedure:

1. Set property EXP.FileName to the name of the file to be created.
2. Set property EXP.ErrorFileName to the export error file.
3. Set property EXP.LogFileName to the export log file.
4. Create a procedure named Export in the app or event. This procedure will execute the Run Export command.
5. Type the following code (in the calling procedure):

```
Run PROCEDURE EXP\Init.lvpro
Run PROCEDURE EXP\OnExport.lvpro

If $EXP.IsValid$
...
END If
```

The EXP.Status object is described below:

Variable[.Property] Name	Type	Value
EXP.Status.Elapsed	STRING	The elapsed time from the log file
EXP.Status.Filtered	NUM	The number of records filtered from the source during export
EXP.Status.MaxError	NUM	The maximum errors settings from the export specification document

Variable[.Property] Name	Type	Value
EXP.Status.Read	NUM	The total number of records read from the export source
EXP.Status.Rejected	NUM	The number of records from the export source that were rejected
EXP.Status.Result	STRING	The result includes one of the following values: ■ **SUCCESS**: if the export was completed successfully ■ **ERROR**: if the maximum number of errors was exceeded ■ **WARNING**: if the export completed with errors not exceeding the maximum number of errors

Parsing the export log

You may generate the export status object even if you are not using the export handling procedure. The export status parses the results of the export log into an object variable.

To generate the export status object:

1. Run the procedure EXP\Init.lvpro.
2. Set the property EXP.LogFileName to the export log file.
3. Run the procedure EXP\ParseStatus.
   The following is the a sample code:

```
Run PROCEDURE EXP\Init.lvpro


Set PROPERTY EXP.logFileName = "<name of log file>"
Run PROCEDURE EXP\ParseStatus.lvpro
```

Using the FORM Code Library

The solutions framework provides a number of code libraries that can be used when developing apps and events. The FORM library provides utilities for working with forms in an app.

Generating a symbol selection form

You can use the FORM code library to generate a form to select symbols from one or more dimensions. The symbol selector requires that you setup variables to define what dimensions selections will be made from and any options related to the dimensions. This process will also create the target variables for the symbol selectors, if they do not exist. The variable name will be the name of the dimension the symbol selector is used for.

To generate a symbol selector:

1. Create a string list variable named selectDimensionList
2. Set selectDimensionList to the names of the dimensions to select
3. For each dimension, set options.
4. Execute procedure FORM\SelectSymbols
5. Check the user response

> **Note:** If no symbol selection is required, the variable FORM_ButtonClicked will be set to OK. No symbol selection is required in a case where: all the required target variables exist, and the variables all have a value. This allows for seamless integration with symbol selection defined in Longview Client.

Example:

```
Create VARIABLE selectDimensionList[] as STRING
Set VARIABLE selectDimensionList = "TIMEPERIODS|ENTITIES"
Run PROCEDURE "FORM\SelectSymbols.lvpro"
If "$FORM_ButtonClicked$" == "OK"
...
END If
```

To set options for a dimension:

1. Restrict symbols: Create and set a RANGE variable named <Dimension>Symbols;
2. Default selection: Create and set a STRING or RANGE variable named <Dimension>Default; ability to select multiple leaf

> **Note:** Use a RANGE variable if the options will allow multiple selections and you want to set a default with multiple symbols selected.

3. Selection options: Create and set a STRING variable named <Dimension>Options. Define the options using 1 or 0 as follows:
   a. 1 to indicate a selection is required
   b. 1 to indicate the user may select a leaf symbol
   c. 1 to indicate the user may select a parent symbol
   d. 1 to indicate the user may select a read-only symbol
   e. 1 to indicate the user may select multiple symbols

> **Note:** The default option if not specified is "11000" (select is required and user may select a leaf symbol).

4. Attribute filter: Create and set a STRING variable named <Dimension>Filter.

Example, with options

This example creates a symbol selection form for: time periods, with the ability to select a single leaf time period and entities with the ability to select multiple leaf entities with USD as the functional currency.

Example with options:

```
Create VARIABLE selectDimensionList[] AS STRING
Set VARIABLE selectDimensionList = "TIMEPERIODS|ENTITIES"
Create VARIABLE TIMEPERIODSDefault AS STRING
Set VARIABLE TIMEPERIODSDefault = "[[SYSTEM,SGPCurrentPeriod,DBDefault]]"
Create VARIABLE TIMEPERIODSSymbols AS RANGE
Set VARIABLE TIMEPERIODSSymbols = "Actual_YTD#99"
Create VARIABLE ENTITIESDefault AS STRING
Set VARIABLE ENTITIESDefault = "E11211"
Create VARIABLE ENTITIESOptions AS STRING
Set VARIABLE ENTITIESOptions = "11001"
Create VARIABLE ENTITIESSymbols AS RANGE
Set VARIABLE ENTITIESSymbols = "ENTITIES_ALL#99"
Create VARIABLE ENTITIESFilter as STRING
Set VARIABLE ENTITIESFilter = "ALL{ZGPNativeCurrency{EQ{USD"
Run PROCEDURE "FORM\SelectSymbols.lvpro"
If "$FORM_ButtonClicked$" == "OK"
...
END If
```

Generating symbol selection controls

You can use the FORM code library to generate controls to select symbols from one or more dimensions that can be appended to an existing form. This allows you to use the code library to create the symbol selectors for use in a custom form.

Using the generator also creates the target variables for the symbol selectors, if required.

To generate symbol selection controls:

1. Follow all the steps outlined to generate a symbol selection form up to the execute step.
2. Execute procedure FORM\GenerateSymbolSelectors.
3. Add the form generated to an existing form
4. Show the form and check the user's selection.

Example

This example shows appending symbol selection to an existing form named Main.

Example:

```
Create DOCUMENT "Form"
Write "Form" FILE "Main.lvfrm"
Create VARIABLE selectDimensionList[] as STRING
Set VARIABLE selectDimensionList = "TIMEPERIODS|ENTITIES"
Run PROCEDURE "FORM\GenerateSymbolSelectors.lvpro"
Write "Form" FILE "SymbolSelectors" APPEND
Show FORM USING "FORM"
Unload "FORM"
Unload "SymbolSelectors"
If "$LVS_ButtonClicked$" == "OK"
...
END If
```

Using the ICON Code Library

The solutions framework provides a library of icons that can be used in apps.

Note:

- Some icons have been deprecated. These icons have been replaced with an equivalent icon from the new set of icons for backwards compatibility, but should not be used going forward.
- The icons used come from Remix Icon - Open source icon library.
  If you need additional icons you can download them from the link above. Icons used in toolbars should have the following properties:
  - Size: 24px
  - Color: #546176

Icon	Location
	ICON\22x22\about.png
	ICON\22x22\add.png
	ICON\22x22\add_to_folder.png
	ICON\22x22\alert.png DEPRECATED: ICON\22x22\alert_triangle_or.png

Icon	Location
	ICON\22x22\alert_filled_red.png
	ICON\22x22\arrow_left.png
	ICON\22x22\arrow_right.png
	ICON\22x22\bookmarks.png
	ICON\22x22\calculator.png
	ICON\22x2\calendar.png
	ICON\22x22\circle_add.png
	ICON\22x22\circle_check.png
	ICON\22x22\circle_check_grn.png
	ICON\22x22\circle_delete.png
	ICON\22x22\circle_delete_red.png
	ICON\22x22\cog.png
	ICON\22x22\collaborate.png
	ICON\22x22\collapse_all.png
	ICON\22x22\comment.png
	ICON\22x22\content.png
	ICON\22x22\copy.png
	ICON\22x22\copy_settings.png
	ICON\22x22\cut.png
	ICON\22x22\data_ok.png
	ICON\22x22\data_unknown.png

Icon	Location
	ICON\22x22\delete.png
	ICON\22x22\down_arrow.png
	ICON\22x22\download.png
	ICON\22x22\edit.png
	ICON\22x22\exit.png
	ICON\22x22\export.png
	ICON\22x22\flash_edit.png
	ICON\22x22\folder.png
	ICON\22x22\folder_open.png
	ICON\22x22\help.png
	ICON\22x22\hierarchy
	ICON\22x22\home
	ICON\22x22\image
	ICON\22x22\import.png
	ICON\22x22\info.png DEPRECATED: ICON\22x22\info_filled_blu.png
	ICON\22x22\note.png DEPRECATED: ICON\16x16\note.png
	ICON\22x22\prefs.png
	ICON\22x22\prefs_unselected.png
	ICON\22x22\refresh.png
	ICON\22x22\save.png

Icon	Location
	ICON\22x22\search.png
	ICON\22x22\select_symbol.png
	ICON\22x22\ticket.png
	ICON\22x22\up_arrow.png
	ICON\22x22\upload.png

Using the IMP Code Library

The solutions framework provides a number of code libraries that can be used when developing apps and events. The IMP library provides utilities for working with import or export specs in an app or event.

IMP global variables

The IMP code library creates and sets a number of variables that are used internally, but can also be used within an app. In most cases these variables will only be directly referenced in a free form app or configuration library. These variables are initialized by procedure IMP\Init.lvpro, which would need to be called within an app in order to use the variables.

Variable Name	Type	Initial Value
IMP_AssignDimension	STRING	
IMP_AssignSymbolList	RANGE	
IMP_ClearData	NUM	0
IMP_Delimiter	STRING	,
IMP_ErrorFileName	STRING	$CORE_LogPath$\Import_Error.txt
IMP_FileName	STRING	
IMP_HasAssignDimensionField	NUM	1
IMP_IsValid	NUM	0
IMP_LogFileName	STRING	$CORE_LogPath$\Import_Log.txt
IMP_MaxErrors	NUM	0
IMP_UseAssign	NUM	0

Using the create import spec utility

The create import spec utility is used to create an import spec that will import a file where each record in the file contains an exact match for symbol sin each dimension. The utility supports both records with single value field or multiple value fields. The import spec document created will be named ImportSpec.

To create an import spec for records with a single value field:

```
Run PROCEDURE IMP\Init.lvpro
Run PROCEDURE "IMP\CreateImportSpec..lvpro"
...
Unload "ImportSpec"
```

To create an import spec for records with multiple values fields (sample assumes time periods):

```
Run PROCEDURE IMP\Init.lvpro
Set VARIABLE IMP_AssignDimension = "TIMEPERIODS"
Set VARIABLE IMP_AssignSymbolList = "<list of time period symbols (one per
value field)>"
Run PROCEDURE "IMP\CreateImportSpec..lvpro"
...
Unload "ImportSpec"
```

To create an import spec for records with a single value, but targeting a different symbol for one dimension (sample assumes time periods):

```
Run PROCEDURE IMP\Init.lvpro
Set VARIABLE IMP_AssignDimension = "TIMEPERIODS"
Set VARIABLE IMP_AssignSymbolList = "<target time period>"
Set VARIABLE IMP_HasAssignDimensionField = 1
Run PROCEDURE "IMP\CreateImportSpec..lvpro"
...
Unload "ImportSpec"
```

Using the import handling procedure

The import handling procedure provides standard error handling and validation for import processes. The import handling procedure performs the following steps:

1. Initializes IMP_IsValid to 1
2. Executes procedure Import.lvpro (app or event specific procedure)
3. Performs error handling as follows:
   - If non-import related error (like a syntax error, or missing file), reports error. IMP_ISValid = 0
   - If import related error, where MaxErrors setting is exceeded: reports error, and allows user to access import log and error file. IMP_IsValid = 0

- If no error, but import errors less than or equal to MaxErrors setting, provides a message, allowing the user to review errors and choose to proceed with upload. IMP_IsValid = 0 (user clicks No), IMP_IsValid = 1 (user clicks Yes).

To use the import handling procedure:

1. Create a procedure named Import in the app or event. This procedure will execute the Run Import command.

2. Type the following code (in the calling procedure):

```
Run PROCEDURE IMP\Init.lvpro
Run PROCEDURE IMP\OnImport.lvpro
If $IMP_IsValid$
...
END If
```

Generating the import status variable

The import status variable summarizes the contents of the import log into an object that can be used as part of the import process.

The following are the details of the IMP_Status object:

Variable (Property) Name	Type	Value
IMP_Status.Elapsed	STRING	The elapsed time from the log file
IMP_Status.Filtered	NUM	The number of records from the import source filtered during an import
IMP_Status.MaxError	NUM	The maximum errors settings from the import specification document
IMP_Status.Read	NUM	The total number of records read from the import source
IMP_Status.Rejected	NUM	The number of records from the import source that were rejected

To generate the import status object:

1. Run the procedure IMP\Init.lvpro.

2. Set the variable IMP_LogFileName to the export log file.

3. Run the procedure IMP\ParseStatus.

The following is the sample code:

```
Run PROCEDURE IMP\Init.lvpro
```

```
Set VARIABLE IMP_LogFileName = "<name of log file>"
Run PROCEDURE IMP\ParseStatus.lvpro
```

Using the LOG Code Library

The solutions framework provides tools for logging messages to a file to track execution at a summary level and to provide additional tracking during debugging. Usually this is used in conjunction with events to create the event log file but can also be used in apps.

Using logging tools

Logging tools provide several different ways to write information to the log file. The log file is stored in memory until logging is complete or an error occurs.

The log file is stored in the following location:

- For apps with Re-use file for log and history unchecked:

```
Documents\Longview\<Longview Identifier>\logs\<AppName>\<Date:
yyyyMMdd>_<Time: hhmmss>\
```

- For apps with Re-use file for log and history checked:

```
Documents\Longview\<Longview Identifier>\logs\<App Name>\
```

Starting the log file

Before logging can occur, the log file must be started. The following will be written to the log file when it is started:

1. The date and time the log was started
2. The framework version
3. For apps the following additional information is written:
   - The app version
   - The template type
   - The template version

To start the log file use the following command:

```
Run PROCEDURE "LOG\Start.lvpro"
```

Note: When debug mode is enabled the log is started as some debug items also write information to the log file.

Ending the log file

The log file is completely in memory until it is ended. The date and time the log ended will be written to the log file.

Note: In the case of an error the log file is saved to disk. If the error terminates the process the log file will not contain the end date and time.

To end the log file use the following command:

```
Run PROCEDURE "LOG\End.lvpro"
```

Note: When debug mode is enabled the log is ended at the end of the process

Logging messages

You can write messages to the log file.

To write a single message to the log file, use the following commands:

```
Set VARIABLE LOG_Message = "Message"
Run PROCEDURE "LOG\Message.lvpro"
```

To write multiple messages to the log file, use the following commands:

```
Set VARIABLE LOG_MessageList = "Message 1"
Set VARIABLE LOG_MessageList = ListAppend(LOG_MessageList, "Message 2")
Run PROCEDURE "LOG\MessageList.lvpro"
```

Inserting blank lines into the log

For readability it is often desirable to leave blank lines in the log file.

To insert a blank line in the log file, use the following command:

```
Run PROCEDURE "LOG\BlankLine.lvpro"
```

Logging variable values

You can write the value of a variable or property to the log file. This can be useful for debugging purposes and also to track what is being executed in events.

To log the value of one or more variables, use the following commands:

```
Set VARIABLE LOG_VariableList = "Variable1"
Set VARIABLE LOG_VariableList = ListAppend(LOG_VariableList, "Variable2")
Run PROCEDURE "LOG\VariableList.lvpro"
```

Appending a file to the log

You can append the contents of an existing file to the log. This can be useful when there is a complex set of information to be logged that is easier to create as a document that to use the primitive log utilities. Use the Save command to save dynamic document to disk prior to using the append file utility.

Note: The file to append must exist on the local disk.

To append the contents of a file to the log:

1. Write the following code in a procedure document:

```
Set VARIABLE CORE_FileName = "$CORE_LogPath$\FileToAppend.txt"
Run PROCEDURE "LOG\AppendFile.lvpro"
```

Using the SYM Code Library

The solutions framework provides a number of code libraries that can be used when developing apps and events. The SYM library provides utilities for working with symbols in application framework.

Resolving a lock specification

The SYM code library provides a utility for resolving a lock specification for a dimension. This utility is useful for automating the task of locking a subset of a query. For example, in an input app that allows the user to use input tools, restricting the lock to cover only symbols that can be modified is essential.

To resolve a lock specification:

1. Set the dimension to resolve the lock specification for (CORE_Dimension)
2. Set the query spec for the dimension ($CORE_Dimension$Query)
3. Set the excluded symbols ($CORE_Dimension$Exclude)
4. Run procedure SYM\ResolveLockSpec.lvpro
5. Use the variable $CORE_Dimension$LockSpec for the symbol in the lock spec for the specified dimension

Example:

```
Set VARIABLE CORE_Dimension = "ACCOUNTS"
Create VARIABLE ACCOUNTSQuery AS RANGE
```

```
Set VARIABLE ACCOUNTSQuery = "OPEXEXDA#99"
Create VARIABLE ACCOUNTSExclude AS RANGE
Set VARIABLE ACCOUNTSExclude = "SALEXP|4035"
Run PROCEDURE "SYM\ResolveLockSpec.lvpro"
```

The variable ACCOUNTSLockSpec will be created and populated with symbol specifications for locking the query area with the specified exclusions.

Working with multiple lock specifications in a single dimension

In some cases you may be working with multiple lock specs and require different subsets of symbols for each lock spec for the same dimension. In this case you will need to create your own variable to hold the specification for each lock spec. Use the ListAppend function to populate the specific variable after executing SYM\ResolveLockSpec.lvpro.

Example:

```
Set VARIABLE CORE_Dimension = "ACCOUNTS"
Create VARIABLE ACCOUNTSQuery AS RANGE
Create VARIABLE ACCOUNTSExclude AS RANGE
//Repeat for each lock spec
Create VARIABLE ACCOUNTSLockSpec1 AS RANGE
Set VARIABLE ACCOUNTSQuery = "OPEXEXDA#99"
Set VARIABLE ACCOUNTSExclude = "SALEXP|4035"
Run PROCEDURE "SYM\ResolveLockSpec.lvpro"
Set VARIABLE ACCOUNTSLockSpec1 = ListAppend(CORE_EmptyList,
ACCOUNTSLockSpec)
```

Note: Using ListAppend with the CORE_EmptyList effectively copies the list into another variable.