MapBasic 11.5.3 Release Notes

This document provides information on new and enhanced features that have been introduced into MapBasic since version 11.5. It also contains sections on resolved customer issues and some remaining known issues that are important for users to be aware of.

United States:
Phone: 518.285.6000
Fax: 518.285.6070
Sales: 800.327.8627
Government Sales: 800.619.2333
Technical Support: 518.285.7283
Technical Support Fax: 518.285.7575
www.pb.com/software

Canada:
Phone: 416.594.5200
Fax: 416.594.5201
Sales: 800.268.3282
Technical Support:.518.285.7283
Technical Support Fax: 518.285.7575
www.pb.com/software

Europe/United Kingdom:
Phone: +44.1753.848.200
Fax: +44.1753.621.140
Technical Support: +44.1753.848.229
www.pitneybowes.co.uk/software

Asia Pacific/Australia:
Phone: +61.2.9437.6255
Fax: +61.2.9439.1773
Technical Support: 1.800.648.899
www.pitneybowes.com.au/software

March 6, 2013

Table of Contents

- **Upgrading MapBasic** 1
- **Advanced Installation for System Administrators** 2
- **Maintenance Release 11.5.3** 2
- **Maintenance Release 11.5.2** 3
- **Maintenance Release 11.5.1** 7

Upgrading MapBasic

Your computer must be connected to the Internet to download the 11.5.3 MapBasic Maintenance Release.

You may install this Maintenance Release provided the following product is already installed:

- MapBasic 11.5, 11.5.1, or 11.5.2.

To upgrade MapBasic to the version 11.5.3 Maintenance Release:

1. Open your internet browser and go to the following URL:
   http://www.pbinsight.com/support/product-downloads/item/mapbasic-v11.5.3-maintenance-release
2. On the web page, click the download link and follow your web browser's instructions for opening and running the patch file.
   (The file is called **MapBasic11.5.3MaintenanceRelease.exe** should you choose to save and run it later.)
3. Follow the instructions to complete the upgrade.

It is important to wait until the installation completes.

After upgrading MapBasic to 11.5.3, any MapBasic programs (.mbx files) that you create require MapInfo Professional version 11.5.3 or later to run. In other words, the users of your .mbx file will need to upgrade their MapInfo Professional installations to 11.5.3 or later to run your .mbx file.

Advanced Installation for System Administrators

This section is for the system administrator installing this Maintenance Release using a command line.

Upgrading in Silent Mode

To run the MapBasic Maintenance Release installer in silent mode, from a command prompt, type:

```
MapBasic11.5.3MaintenanceRelease.exe /s /v"SILINST=True"
```

Where:

`/s` = runs the setup.exe silently.

`/v` = passes the parameter "`SILINST=True`" into the setup.exe to suppress the GUI.

A progress dialog may display during the installation.

Windows 7, **Server 2008**, and **Server 2008 R2** users may see a prompt for permission to continue. Click **Allow** or **OK** to proceed.

Maintenance Release 11.5.3

This section lists the updates made since the 11.5.2 release. The 11.5.3 Maintenance Release is cumulative and includes all updates, including the function and statement enhancements, supplied with the 11.5.1 and 11.5.2 Maintenance Release.

- Set Table Statement Enhancement
- TableInfo( ) Funtion has a New Attribute
- Progress and Resolution of Outstanding Issues

Set Table Statement Enhancement

The **Set Table** statement controls settings that affect how and whether a table can be edited. You can this statement to flag a table as read-only (so that the user will not be allowed to make changes to the table). You can also use **Set Table** to activate or de-activate special editing modes which disable safety mechanisms for the sake of improving editing performance.

This statement has a new, optional clause: **Persist**. You can set this clause to either **On** or **Off**. The change has been underlined in the following syntax.

Syntax

```
Set Table tablename
   [ FastEdit { On | Off } ]
   [ Undo { On | Off } ]
   [ ReadOnly ]
   [ Seamless { On | Off } [ Preserve ] ]
   [ UserMap { On | Off } ]
   [ UserBrowse { On | Off } ]
   [ UserClose { On | Off } ]
   [ UserEdit { On | Off } ]
   [ UserRemoveMap { On | Off }} ]
   [ UserDisplayMap { On | Off } ]
   [ Persist { On | Off } ]
   [ datum datum_number ]
```

Description

The **Persist Off** clause marks a table, so that it will not be written to the workspace when a workspace is saved. The **Persist** or **Persist On** clause marks a table, which was previously marked as **Persist Off**, so that it will be written back to the workspace when a workspace is saved.

For more information on the **Set Table** statement, refer to the *MapBasic Reference.*

TableInfo( ) Funtion has a New Attribute

The TableInfo( ) function returns information about an open table. This function now includes a new attribute code, as shown below:

Attribute Code	ID	TableInfo( ) returns
TAB_INFO_PERSIST	44	Logical result: TRUE if the table should be persisted to the workspace.

Progress and Resolution of Outstanding Issues

Issue Number	Description and Resolution
MIPRO-32886	The CommandInfo( ) function does not always return a coordinate in the correct coordinate system. In MapBasic 11.5.2, some applications encountered an issue involving a Custom Point Tool (DM_CUSTOM_POINT) with CommandInfo(CMD_ INFO_X) or CommandInfo(CMD_INFO_Y). If a mouse click event handler sets the coordinate system and then tries to retrieve the coordinates using CommandInfo(CMD_ INFO_X) or CommandInfo(CMD_INFO_Y), the coordinates may not return the correct coordinate system. The issue occurred due to the mouse click event handler calling the function CommandInfo(CMD_INFO_TOOLBTN) prior to changing the coordinate system. **Resolution:** Resolved. The CommandInfo( ) function now return coordinates in the current coordinate system.

Maintenance Release 11.5.2

This section lists the updates made since the 11.5.1 release. The 11.5.2 Maintenance Release is cumulative and includes all updates, including the function and statement enhancements, supplied with the 11.5.1 Maintenance Release.

- **Set Designer Legend Statement Controls Anti-Aliasing for Legend Symbols**
- **TableInfo( ) Function has New Attributes**
- **WFS Refresh Table Statement Enhancement**
- **Register Table Statement Enhancement**
- **Server Link Table Statement Enhancement**
- **Set Browse Statement Enhancement**
- **New WKTToCoordSysString$( ) function**
- **Progress and Resolution of Outstanding Issues**

Set Designer Legend Statement Controls Anti-Aliasing for Legend Symbols

The Set Designer Legend statement refreshes the styles shown in each cartographic legend frame and resets the orientation to portrait or landscape for an existing cartographic legend created with the **Create Designer Legend** statement.

Symbols are now drawn as vector. Line and region styles must still be drawn as raster, but are no longer anti-aliased, so they will not appear blurry. By default, anti-aliasing is set to **off** for the Set Designer Legend statement.

To turn **on** anti-aliasing for all currently open and any new Legend Designer windows:

```
   Set Designer Legend Antialias On
```

To turn **off** anti-aliasing, use:

```
   Set Designer Legend Antialias Off
```

To set anti-aliasing to be on as the default setting for legends, consider adding this MapBasic command to a workspace, such as startup.wor, or to the MapBasic window at runtime.

TableInfo( ) Function has New Attributes

The TableInfo( ) function returns information about an open table.This function now includes new attribute codes:

Attribute Code	ID	TableInfo( ) returns
TAB_INFO_ADSK_TEXTOBJECT	42	Logical result: TRUE if the table is an Autodesk text table.
TAB_INFO_OVERRIDE_COORDINATE_ ORDER	43	Logical result: TRUE if the table is a Web Feature Service (WFS) table or a Web Map Service (WMS) table with the coordinate order override turned on.

Examples

To determine if a Web Feature Service (WFS) table has the coordinate order override set, use the TAB_INFO_ OVERRIDE_COORDINATE_ORDER (43) attribute. This returns TRUE if the table is a WFS table with the coordinate order override turned on and it returns FALSE otherwise.

```
TableInfo(MyWFSTable, TAB_INFO_OVERRIDE_COORDINATE_ORDER)
```

WFS Refresh Table Statement Enhancement

The WFS Refresh Table statement refreshes a Web Feature Service (WFS) table from the server. This statement has a new, optional, clause for Override Coordinate Order.

For more details about how MapInfo Professional 11.5.2 supports WFS 1.1.0 by overriding a coordinate order, see the *MapInfo Professional 11.5.2 Release Notes*.

Syntax

```
WFS Refresh Table alias
   [ Using Map [ Window window_id ] ]
   [ Override Coordinate Order { On | Off } ]
```

Description

The **Override Coordinate Order** clause is applied when the coordinate order is incorrect for only some tables retrieved from a server. This clause applies a coordinate order override at the table level (instead of at the server level). This clause can be used in conjunction with the **Using Map** clause.

For an example of how to determine if a Web Feature Service (WFS) table has the coordinate order override set, see the examples under **TableInfo( ) Function has New Attributes**.

Register Table Statement Enhancement

The Register Table statement turns a spreadsheet, database, text file, raster, or grid image into a MapInfo Professional table. This statement checks a non-native file, for example, a dBASE file, and builds a TAB file. Only then you can access the file as a MapInfo Professional table.

The Register Table statement does not change the non-native source file. It determines the data type of the columns in the file, and creates the TAB file. To open the new table, you must use the **Open Table** statement.

Each file needs to be registered only once.

This statement has new, optional, style clauses (underlined in the following syntax description).

Syntax

```
Register Table source_file {
   Type ODBC
      Connection { Handle connection_number | connection_string }
      Toolkit toolkit_name
      Cache { ON | OFF }
      [ Autokey { ON | OFF }]
      Table SQLQuery
      [ Versioned { ON | OFF }]
      [ Workspace Workspace_name ]
      [ ParentWorkspace ParentWorkspace_name ]
      [ Symbol ...] [Linestyle Pen (...)]
      [ Regionstyle Pen (...) Brush (...)]
   [ ReadOnly ]
}
```

Description

The following new clauses are applied when the source file is of type ODBC:

- The **Symbol** clause specifies a symbol style for point objects.
- The **Brush** clause specifies a fill style for graphic objects.
- The **LineStyle Pen** clause specifies a line style for line object types.
- The **Regionstyle Pen(…) Brush(…)** clause specifies the line style and fill style for region object types.

A second enhancement has been made that applies to all source files: the **ReadOnly** clause indicates that the table cannot be edited.

Server Link Table Statement Enhancement

You can use the Server Link Table statement to create a linked MapInfo table on disk. A linked table cannot be packed. You can also use this statement to establish a connection to a database server and link a table. A linked table identifies the remote data to be updated, which is stored as metadata in the TAB file.

This statement has new, optional, style clauses (underlined in the following syntax description).

Syntax 1

```
Server Link Table
   SQLQuery
   Using ConnectionString
   [ Symbol ...] [Linestyle Pen (...)]
   [ Regionstyle Pen (...) Brush (...)]
   Into TableName
   Toolkit Toolkitname
   [ File FileSpec ]
   [ ReadOnly ]
   [ Autokey { Off | On }]
```

Syntax 2

```
Server ConnectionNumber Link Table
   SQLQuery
   Toolkit toolkitname
   [ Symbol ...] [Linestyle Pen (...)]
   [ Regionstyle Pen (...) Brush (...)]
   Into TableName
   [ File FileSpec ]
   [ ReadOnly ]
   [ Autokey { Off | On }]
```

Description

The **Symbol** clause specifies a symbol style for point objects.

The **Brush** clause specifies a fill style for graphic objects.

The **Linestyle Pen** clause specifies a line style for line object types.

The **Regionstyle Pen(…) Brush(…)** clause specifies the line style and fill style for region object types.

Set Browse Statement Enhancement

You can now use the Set Browse statement to sort up to five columns using the **Order By** clause.

Syntax

```
Set Browse
   [ Window window_id ]
   [ Grid { On | Off } ]
   [ Row row_num ]
   [ Column column_num ]
   [ Columns Resize ]
   [ Order By sortColumn [ Desc ] [, sortColumn2 . . .] ]
   [ Order None ]
   [ Filter Where
         (filterCondition [ And | Or  filterCondition ] )
      [ And (filterCondition [ And | Or filterCondition ] ) ...] ]
   [ Filter None ]
   [ SortFilter { On | Off }]
   [ Reapply ]
```

Example

```
Set Browse Order By Country, State, City, ZipCode, IncomeGroup
```

New WKTToCoordSysString$( ) function

Purpose

Converts a Well-Known Text (WKT) string into a MapBasic coordinate system (CoordSys) clause. The **CoordSys** clause specifies the coordinate system used by the paper map. For more details, see **CoordSys** clause. You can call this function from the MapBasic Window in MapInfo Professional.

Syntax

```
WKTToCoordSysString$( wkt_string )
```

wkt_string is a Well-Known Text (WKT) string value.

Return Value

String expression, representing a coordinate system. If no string value is found, returns an empty string.

Example

The following example:

```
print WKTToCoordSysString$("GEOGCS["+"""NAD27 Latitude/
Longitude,Degrees"""+",DATUM["+"""North_American_Datum_1927"""+",SPHEROID["+"""Clarke
-
1866"""+",6378206.4,294.9786982139006],AUTHORITY["+"""EPSG"""+","+"""6267"""+"]],PRIME
M["+"""Greenwich"+""",0],UNIT["+"""degree"""+",0.0174532925199433]]")
```

Produces the following string:

```
CoordSys Earth Projection 1, 62
```

See Also:

CoordSys clause, CoordSysStringToWKT$( ) function, Set CoordSys statement

Progress and Resolution of Outstanding Issues

Issue Number	Description and Resolution
MIPRO-16167	If the `Create Table` statement contains a **Using** clause and a **Type Native** clause, the complier will return an error. **Resolution:** Fixed.
MIPRO-22232	The **CurTime( )** and **CurDateTime( )** functions return milliseconds as 000 in the output, when executed in MapBasic. The data types Time and DateTime can store time in milliseconds. However, the **CurTime( )** and **CurDateTime( )** returned the current time with the millisecond value truncated and were only accurate to the second. **Resolution:** Fixed.
MIPRO-25129	The **Sort** dialog box and the MapBasic `Set Browse Order By` statement did not support 5–column sorting. **Resolution:** Fixed.
MIPRO-30385	If a MapBasic program includes a syntax error that is followed by a `Set Browse` statement, the MapBasic compiler (version 11.0 or 11.5) might not report the syntax error. **Resolution:** Fixed.

Maintenance Release 11.5.1

This section lists the updates, including the function and statement enhancements, made since MapBasic 11.5.

- **Set Window Statement Hides the Toolbar on a Legend Designer Window**
- **LegendFrameInfo( ) Function Enhancements**
- **LegendStyleInfo( ) Function Enhancements**
- **Create Designer Legend Statement Enhancements**
- **Alter Designer Frame Statement Enhancements**
- **Create Designer Legend & Alter Designer Frame Custom Order Options**
- **Progress and Resolution of Outstanding Issues**

Some of the updates made in 11.5.1 are specific to working with map legends. MapInfo Professional 11.5 replaces the cartographic legend window with a new and more usable window, called the Legend Designer. The older Create Legend wizard that would create cartographic legends is still available for maintaining maps that pre-date version 11.5. However, as of version 11.5, functions and statements may not fully support the older cartographic legends.

Set Window Statement Hides the Toolbar on a Legend Designer Window

The Set Window statement changes the size, position, title, or status of a window, and controls the printer, paper size, and margins used by MapInfo Professional.

This statement can now be used to hide the toolbar at the top of the Legend Designer window using the following syntax:

```
Set Window windowId Toolbar Off
```

LegendFrameInfo( ) Function Enhancements

The LegendFrameInfo( ) function returns information about a frame within a legend.

The following attributes are updated to work with thematic legend frames, in addition to Map Legend Frames, in the **Legend Designer** window:

Attribute Code	ID	LegendFrameInfo( ) Return Value
FRAME_INFO_TITLE_FONT	9	Returns the font of a legend frame title. If the frame has no title, returns the default title font. *Previously, this would return the default font (Arial,0,10,0) for a thematic legend frame title instead of the font used.*
FRAME_INFO_SUBTITLE_FONT	11	Same as FRAME_INFO_TITLE_FONT (9)
FRAME_INFO_NUM_STYLES	13	Returns the number of styles in a frame. *Previously, this would return zero (0) for a theme frame*.

The following attribute is new. It returns values for both map and thematic legend frames in a Legend Designer window.

Attribute Code	ID	LegendFrameInfo( ) Return Value
FRAME_INFO_NUM_VISIBLE_ ROWS	18	Returns the number of visible rows in a legend frame. For Cartographic Legend windows (prior to version 11.5), returns -1.

LegendStyleInfo( ) Function Enhancements

The LegendStyleInfo( ) function returns information about a style item within a legend frame.

This function now returns style information for thematic legend frames, in addition to map legend frames, in the **Legend Designer** window. As a result, the following attributes now return information for thematic legend frames:

Attribute Code	ID	LegendStyleInfo( ) Return Values
LEGEND_STYLE_INFO_TEXT	1	Returns the text of the style.
LEGEND_STYLE_INFO_FONT	2	Returns the font of the style.

Attribute Code	ID	LegendStyleInfo( ) Return Values
LEGEND_STYLE_INFO_OBJ	3	Returns the object of the style. For legend theme type, possible values are: • Ranged theme – Rectangle, Line or Point • Bar Theme – Rectangle • Grid theme – Rectangle • Graduated Theme – Point • DotDensity theme – Rectangle • Pie theme – Rectangle
LEGEND_STYLE_INFO_ROW_ VISIBLE	4	Returns whether the style row is visible in the legend. For Cartographic Legend windows (prior to version 11.5), returns true.

Example

The following example highlights how to call the LegendStyleInfo( ) function to only return style information for thematic frame styles in the Legend Designer window. This example assumes that the Legend Designer window is the front-most window. It obtains style information from a Ranged Theme Frame #1, Style Sample 1.

```
dim objStyle as object
dim tBrush as Brush
dim tPen as Pen
dim tSymbol as Symbol
dim tFont as Font
dim strSampleText as string

objStyle = LegendStyleInfo(FrontWindow(), 1, 1, LEGEND_STYLE_INFO_OBJ)
tFont = LegendStyleInfo(FrontWindow(), 1, 1, LEGEND_STYLE_INFO_FONT)
strSampleText = LegendStyleInfo(FrontWindow(), 1, 1, LEGEND_STYLE_INFO_TEXT)

Do Case ObjectInfo(objStyle, OBJ_INFO_TYPE)
   Case OBJ_TYPE_POINT
      tSymbol = ObjectInfo(objStyle, OBJ_INFO_SYMBOL)
      Print tSymbol
```

'or use StyleAttr() to return specific properties of the Symbol Style, for example: StyleAttr(tSymbol, SYMBOL_ FONT_NAME)

```
   Case OBJ_TYPE_LINE
      tPen = ObjectInfo(objStyle, OBJ_INFO_PEN)
      Print tPen
```

'or use StyleAttr() to return specific properties of the Pen\Line Style, for example: StyleAttr(tPen, PEN_COLOR)

```
   Case OBJ_TYPE_RECT
      tBrush = ObjectInfo(objStyle, OBJ_INFO_BRUSH)
      Print tBrush
```

'or use StyleAttr() to return specific properties of the Brush\Region Style, for example: StyleAttr(tBrush, BRUSH_ FORECOLOR)

```
   Case Else
      Note "Unexpected Object type"
End Case

Print tFont  'for example: Font ("Arial",0,8,0,0)
```

'or use StyleAttr() to return specific properties of the Font Style, for example: StyleAttr(tFont, FONT_NAME)
'print text of first thematic range value.

```
Print strSampleText    'e.g.: 5,700,000 to 23,700,000
```

Create Designer Legend Statement Enhancements

The Create Designer Legend statement creates a new Legend Designer window to display map style frames for map layers and theme legend frames for thematic map layers for an active Map window. It now includes a new **Order** clause and a new **Display** clause, which are underlined in the following syntax description.

Syntax

```
Create Designer Legend
   [ From Window map_window_id ]
   [ Behind ]
   [ Position ( x, y ) [ Units paper_units ] ]
   [ Width win_width [ Units paper_units ] ]
   [ Height win_height [ Units paper_units ] ]
   [ Window Title { legend_window_title }
   [ Portrait | Landscape | Custom ]
   [ Default Frame Title { def_frame_title } [ Font... ] } ]
   [ Default Frame Subtitle { def_frame_subtitle } [ Font... ] } ]
   [ Default Frame Style { def_frame_style } [ Font... ] } ]
   [ Default Frame Line Width width [ Units paper_units ] ]
   [ Default Frame Region Width width [ Units paper_units ] ]
   [ Default Frame Region Height height [ Units paper_units ] ]
   Frame From Layer { map_layer_id | map_layer_name
      [ Using
         [ Column { column | Object } [ FromMapCatalog { On | Off }]]
         [ Label { expression | Default } ]
      [ Position ( x, y ) [ Units paper_units ] ]
      [ Title { frame_title [ Font... ] }
      [ SubTitle { frame_subtitle [ Font... ] } ]
      [ Columns number_of_columns ] | [ Height frame_height [ Units paper_units ] ]
      [ Order { Default | Ascending | Descending | { Custom id | id : id [, id | id : id
... ] } }
      [ Style [ Font...] [ Norefresh ] [ Text { style_name } [ Display { On | Off } ]
         { Line Pen... | Region Pen... Brush...| Symbol Symbol... } |
            Collection [ Symbol... ]
      [ Line Pen... ] [ Region Pen... Brush ...] } ]
   [ , ... ]
```

Description

The **Order** clause adds the ability to sort or customize the order of rows in map legends. You can sort map legends by style label or by defining your own order. The sort order options are **Default**, **Ascending**, **Descending**, or **Custom**.

A **Default** sort order is the order the Create Legend wizard creates the legend rows. If you create a legend based on unique styles, this will be the order of those styles, which then display as rows in the map legend.

If specifying **Custom** sort order, the *id* values are row ids starting with 1, moving from top to bottom. When looking at the list of rows in the Legend Frame Properties dialog box, the first row in the list has id equal to 1 and so on down the list. For more details, see **Create Designer Legend & Alter Designer Frame Custom Order Options**.

The **Display** clause controls the display of each row. Each row in MapBasic starts with the Style clause. At the end of each Style clause is the optional Display clause. The **Display On** clause makes the row visible. The **Display Off** clause makes the row invisible. If the Display clause is absent, then the row displays.

The **Shade** and **Set Legend** statements still create and modify thematic legends, but only the **Create Designer Legend** statement adds thematic legends to a Legend Designer window (by specifying the theme layer ID using the **Frame From Layer** *id* clause).

The **Columns** or **Height** clauses apply to map and thematic legends. However, all other thematic legend properties must be specified using the **Set Legend** statement.

Workspace Persistence Changes

The **Display** clause is only written to a workspace file for styles that are not visible in a legend frame. If the clause needs to be written, then the workspace version increases to MapInfo Professional 1151.

The **Order** clause is only written to a workspace if the map legend is sorted ascending, descending or custom. If the clause is written then the version increases to 1151.

Alter Designer Frame Statement Enhancements

The Alter Designer Frame statement changes a frame(s) position, title, subtitle, and style of an existing legend created with the Create Designer Legend statement. (To change the size, position or title of the Legend window, use the Set Window statement.) It now includes a new **Order** clause and a new **Display** clause, which are <u>underlined</u> in the following syntax description.

Syntax

```
Alter Designer Frame [ Window legend_window_id ]
   Id { frame_id }
   [ Position ( x, y ) [ Units paper_units ] ]
   [ Title [ frame_title ] [ Font... ] ]
   [ SubTitle [ frame_subtitle ] [ Font... ] ]
   [ Columns number_of_columns ] | [ Height frame_height [ Units paper_units ] ]
   [ Order { Default | Ascending | Descending | { Custom id | id : id [, id | id : id ...
] } }
   [ Style [ Font... ]
      [ ID { id } Text { style_name } [ Display { On | Off } ] {
          [ Line Pen... | Region Pen... Brush... | Symbol Symbol... ] |
           Collection [ Symbol... ] [ Line Pen... ] [ Region Pen... Brush...]
      } ]
   ]
   [ , ... ]
```

Description

The **Order** clause adds the ability to sort or customize the order of rows in map legends. You can sort map legends by style label or by defining your own order. The sort order options are **Default**, **Ascending**, **Descending**, or **Custom**.

If the clause is written, then the version increases to 1151.

A **Default** sort order is the order the Create Legend wizard creates the legend rows. If you create a legend based on unique styles, this will be the order of those styles, which then display as rows in the map legend.

If specifying **Custom** sort order, the *id* values are row ids starting with 1, moving from top to bottom. When looking at the list of rows in the Legend Frame Properties dialog box, the first row in the list has id equal to 1 and so on down the list. See **Create Designer Legend & Alter Designer Frame Custom Order Options** for more details.

The **Display** clause controls the display of each row. Each row in MapBasic starts with the Style clause. At the end of each Style clause is the optional Display clause. The **Display On** clause makes the row visible. The **Display Off** clause makes the row invisible. If the Display clause is absent, the row displays.

The **Columns** or **Height** clauses apply to map and thematic legends. However, all other thematic legend properties must be specified using the **Set Legend** statement.

Create Designer Legend & Alter Designer Frame Custom Order Options

Both the Create Designer Legend statement and the Alter Designer Frame statement now include an **Order** clause with the option of specifying a **Custom** sort order.

```
   Custom id | id : id [, id | id : id ... ]
```

Where:

```
Id1:Id2
```

specifies a range of row ids in increasing order ($Id_2 > Id_1$).

Reordering a List

The syntax for custom order of legend rows is similar to the Set Map statement's **Order** clause (to reorder layers). It is fairly easy to use when you want to reorder near the beginning of a list, but not so easy when you want to reorder near the end. For instance, if you want to reverse the order of the first three rows you only have to use:

```
Order Custom 3, 2, 1
```

You can leave out the rest of the rows. If you have 10 rows and want to switch the last two, you have to list all the ids like this:

```
Order Custom 1, 2, 3, 4, 5, 6, 7, 8, 10, 9
```

To make this more compact, use the following syntax:

```
Order Custom 1:8, 10, 9
```

Indicating a Range of Values

Use a colon (**:**) to indicate a range of values, such as:

Long form:

```
Order Custom 2, 5, 6, 7, 8, 9, 10, 1, 3, 4
```

Short form:

```
Order Custom 2, 5:10, 1, 3, 4
Order Custom 2, 5:10, 1, 3:4 (same as above but also valid)
Order Custom 2, 5:10, 1 (same as above but also valid)
```

Long form:

```
Order Custom 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 12, 13, 14, 15, 16, 17, 18, 19, 20, 11
```

Short form:

```
Order Custom 1:10, 12:20, 11
```

The list of values cannot have duplicates—that will cause an error. For example:

```
Order Custom 1:5, 8, 4:7 // causes an error
```

causes an error, because row ids 4 and 5 are duplicates. To see this, expand the syntax:

```
Order Custom 1, 2, 3, 4, 5, 8, 4, 5, 6, 7 // causes an error
```

The alternate syntax can be used when creating or altering a legend in the Legend Designer window. For workspaces, the short syntax is used when legends in the Legend Designer window have more than 50 rows with a custom order.

Progress and Resolution of Outstanding Issues

Issue Number	Description and Resolution
MIPRO-25119	`If CommandInfo(CMD_INFO_DLG_OK)=TRUE Then...` is run, the expected results are not returned. If the routine is replaced with the implicit logical `If CommandInfo(CMD_INFO_DLG_OK) Then` which removes the `=TRUE` section, then this issue no longer appears. **Resolution:** Fixed.
MIPRO-25253	Legend Designer causes an error on exit from the Integrated Mapping application. **Resolution:** Fixed.