User Manual

Recipe Database

This guide walks through basic information and usage of EasyBuilder Pro Recipe Database.

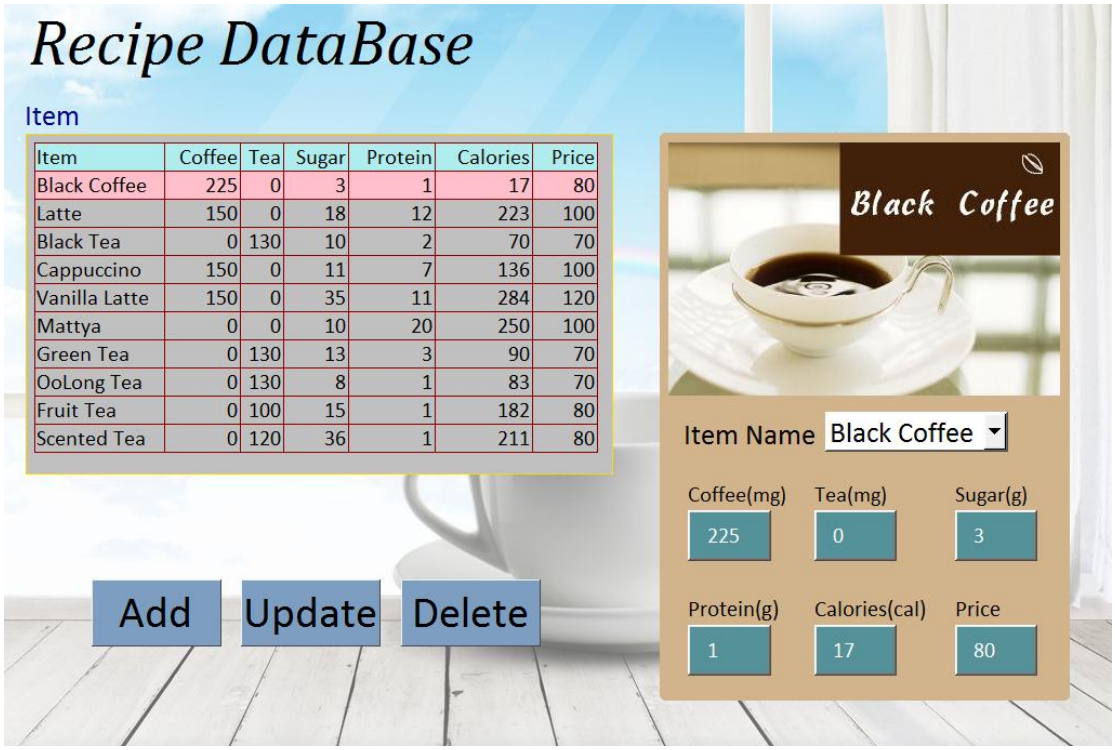
UM014005E_20211215

Table of Contents

1. Introduction of Recipes

1.1. Overview

Recipe DataBase optimized the way of using and editing recipes (RW, RW_A). Recipe DataBase displays the edited recipes in table form, and there's no need to calculate the interval between addresses. Certain Macro functions are provided for searching recipes faster and easier. The process is to build the needed data type in [Recipe Database], enter the values in [Data] tab, and then see the result in [Recipe View]. The recipe data can be used in other objects.



Features

- Displays recipe data in table form.
- Sorts the records in desired column.
- Searches recipe data by Macro.

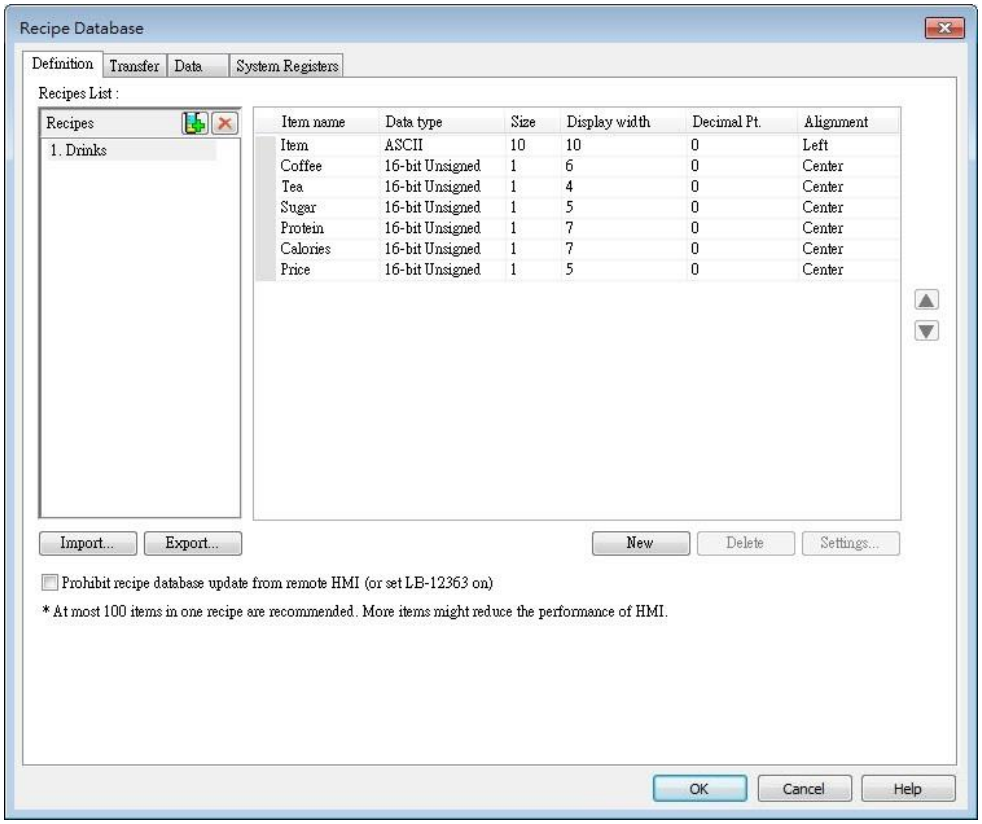
Demo Project

See Recipe_Database Demo Project for the examples presented in this document.

1.2. Features and Objects

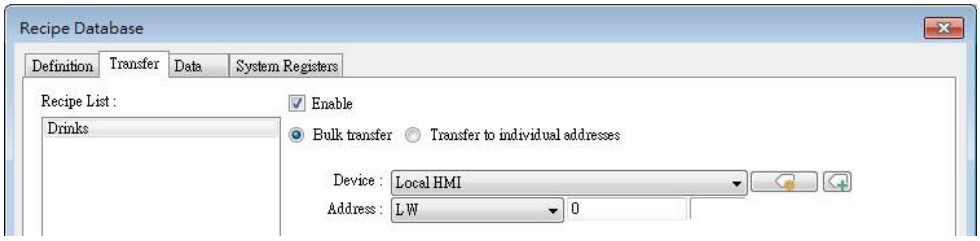
- Definition

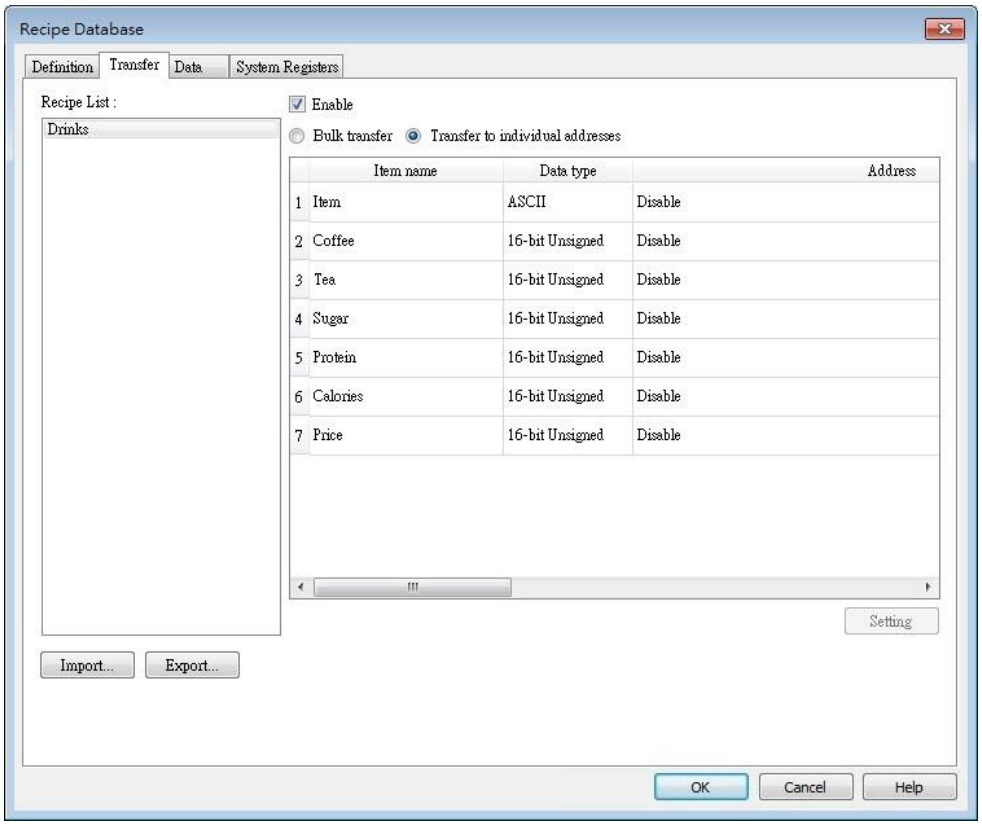
In [Data/History] » [Recipe Database] » [Definition] tab, add recipes in [Recipes List], and specify item names to define the columns in database.



- Transfer (cMT / cMT X Series)

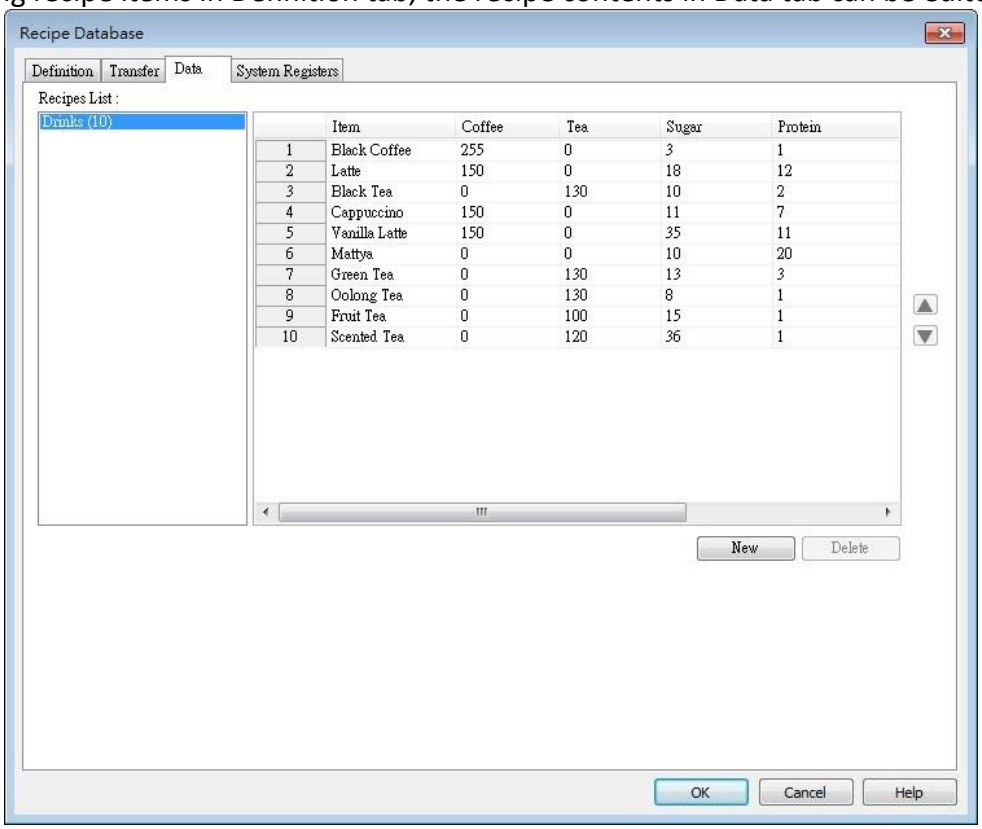
In [Data/History] » [Recipe Database] » [Transfer] tab, the settings for transferring records between the recipe database and the designated destinations can be found. Enter command 5 to write the selected record to the designated destination, or enter command 6 to update the selected record from the source. In Bulk Transfer mode, the transfer destinations of a record are consecutive addresses starting from the designated address. Under this mode, the data type settings of destination addresses must follow those of recipe items in Definition tab. In [Transfer to individual addresses] mode, transfer destination of each item within a record is configured individually.





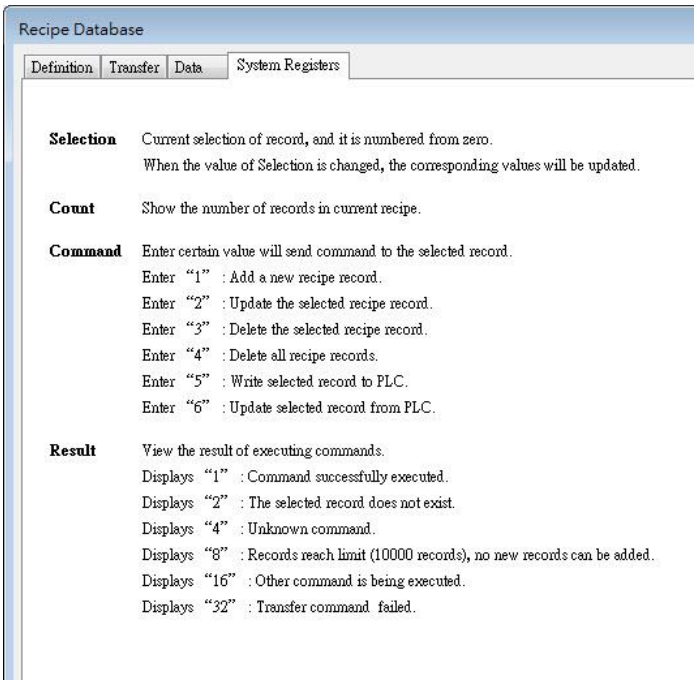
- Data

Open [Data/History] » [Recipe Database] » [Data] tab to edit recipe contents. After setting recipe items in Definition tab, the recipe contents in Data tab can be edited.



- System Registers

  Explanation about Recipe related system registers can be found in this tab.



- Recipe View

Find [Data/History] » [Recipe View]. Recipe View object is used for displaying a specific recipe. Users can view all items and values of a recipe by using this object, or use relevant registers to monitor or modify recipe records.

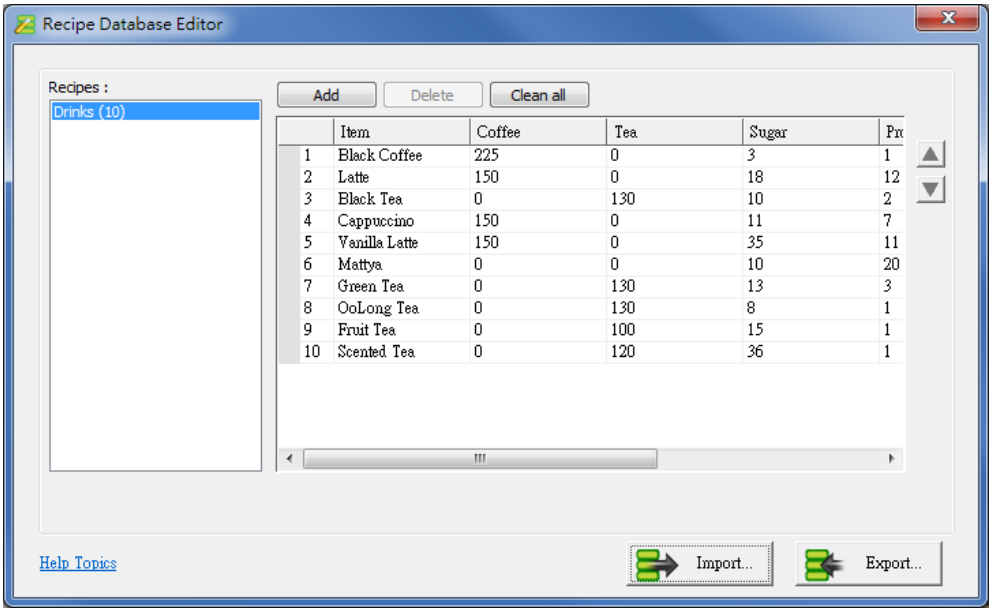
Item	Coffee	Tea	Sugar	Protein	Calories	Price
Black Coffee	225	0	3	1	17	80
Latte	150	0	18	12	223	100
Black Tea	0	130	10	2	70	70
Cappuccino	150	0	11	7	136	100
Vanilla Latte	150	0	35	11	284	120
Mattya	0	0	10	20	250	100
Green Tea	0	130	13	3	90	70
OoLong Tea	0	130	8	1	83	70
Fruit Tea	0	100	15	1	182	80
Scented Tea	0	120	36	1	211	80

- Import/Export

Open [Data/History] » [Import/Export] to import or export recipe database into USB disk, SD card, cMT Series HMI (Importing recipe data is only supported on cMT / cMT X Series models).

- Recipe Database Editor

Open [Tool] » [Recipe Database Editor] to import a *.db file and start editing. This tool allows editing recipe without opening EasyBuilder Pro.



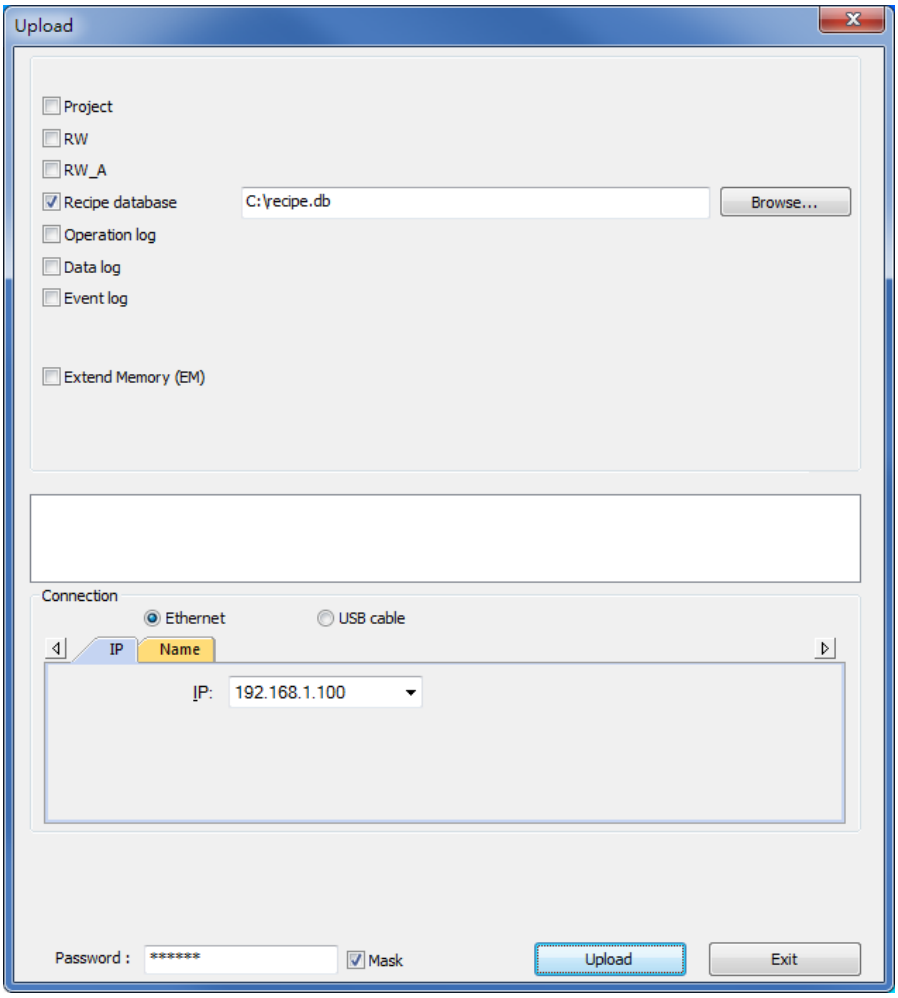
- Recipe Query Functions

  Some Macro functions can be used to query recipe data:

  1. RecipeGet Data: Get recipe data.
  2. RecipeQuery: Query recipe data to obtain the number of records that meets the specified condition.
  3. RecipeQueryGetData: From the result gained by RecipeQuery, get the data of the specific item.
  4. RecipeQueryGetRecordID: From the result gained by RecipeQuery, get the specific record ID.
  5. RecipeSetData: Write data to Recipe Database.

1.3. Upload/Download Recipe Database

Utility Manager offers an option to upload / download recipes. The way is the same as uploading or downloading project files. To upload, click [Utility Manager] and [Upload], select HMI, and then select [Recipe database] check box.



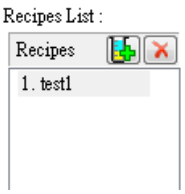
2. How to Build Recipes

2.1. Recipe Settings

In [Data/History] » [Recipe Database] » [Definition] tab, add recipes in [Recipes List], and specify item names to define the columns of database.

Recipes List

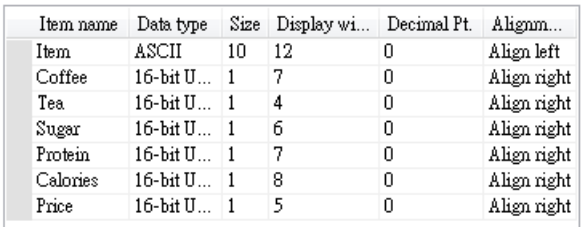
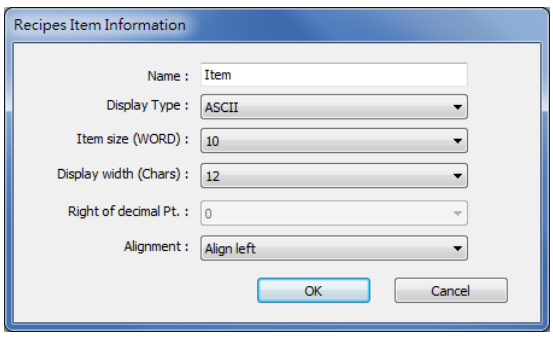
Add or delete recipes in this list.



- Recipe name can't be repeated.
- Only support alphanumeric names.
- Up to 100 recipes allowed.

Recipe Items

Set the data type, size, display width, decimal point, and alignment to display the recipe items. Up to 1000 recipe items allowed.

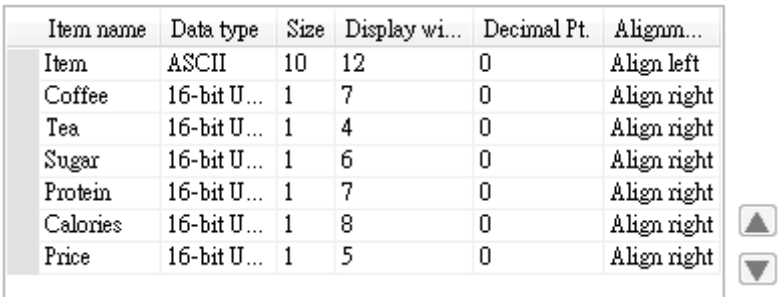
Setting	Description
New	Point to an item and click [New], an identical item will be added. 
Settings	Point to an item and click [Settings], the detailed item information is shown and allows users to modify the contents. 
Delete	Delete an existing recipe item.
Import/Export	Import/Export recipe parameter settings file. The file includes parameters such as item name, data type, size,

	etc. The file can be exported as *.rdef file and then imported to EasyBuilder.

Note

- Each recipe database can hold up to 2000 words and data exceeding this limit cannot be compiled.

Item Settings

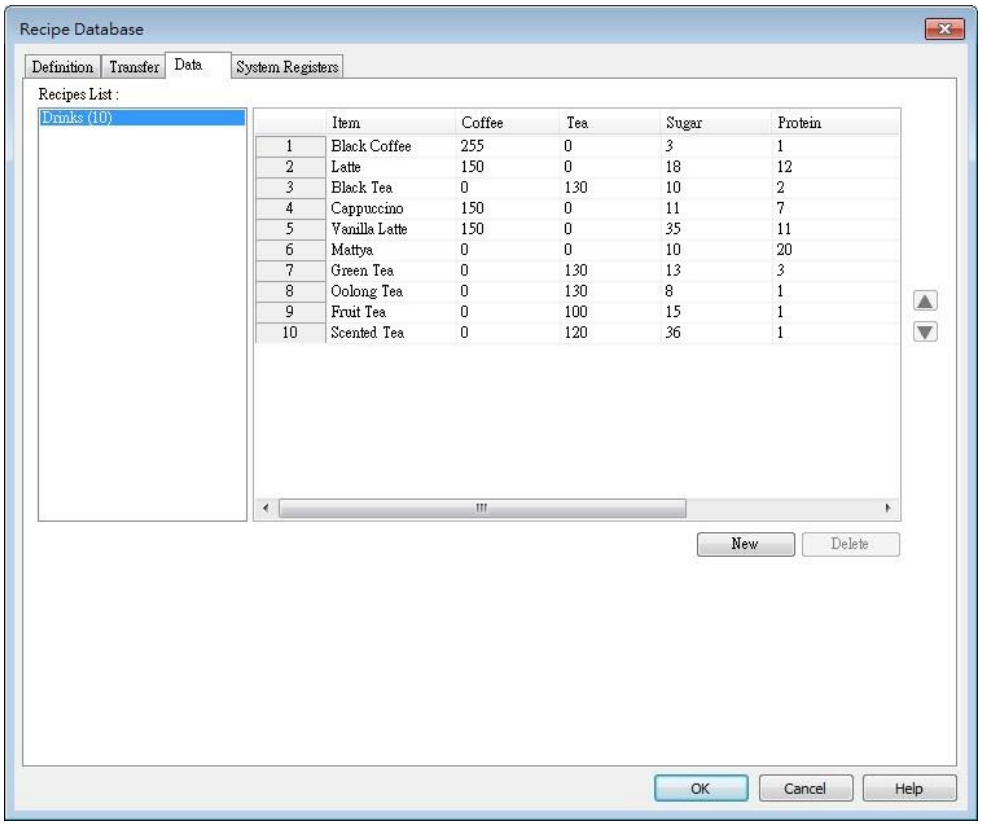


Item name	Data type	Size	Display wi...	Decimal Pt.	Alignm...
Item	ASCII	10	12	0	Align left
Coffee	16-bit U...	1	7	0	Align right
Tea	16-bit U...	1	4	0	Align right
Sugar	16-bit U...	1	6	0	Align right
Protein	16-bit U...	1	7	0	Align right
Calories	16-bit U...	1	8	0	Align right
Price	16-bit U...	1	5	0	Align right

Setting	Description
Item Name	Specifies the item name. Only allows alphanumeric names and "_" symbol.
Data type	The supported data types: 16-bit BCD, 32-bit BCD, 16-bit Hex, 32-bit Hex, 16-bit Binary, 32-bit Binary, 16-bit Unsigned, 32-bit Unsigned, 16-bit Signed, 32-bit Signed, 32-bit Float, ASCII, Unicode, High/Low Reversed, 14 types in total.
Size	Specifies the data length. The data length can only be specified in ASCII, Unicode, High/Low Reversed formats. The limit is 255 words.
Display Width	Specifies the column width in [Recipe View] object.
Decimal Points	Adjusts the number of digits after the decimal point.
Alignment	Aligns recipe data when display them in [Recipe View] object.

2.2. Recipe Database Data

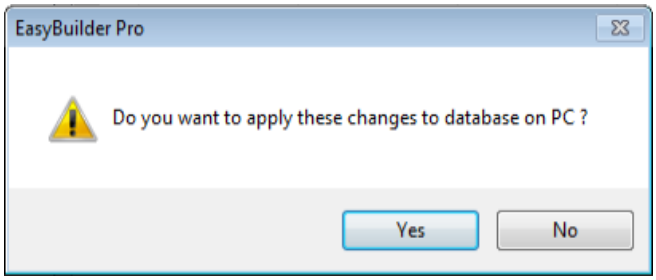
Open [Data/History] » [Recipe Database] » [Data] tab to edit recipe contents. After setting recipe items in Definition tab, the recipe contents in Data tab can be edited.



Setting	Description
Recipes List:	The recipes created in Definition tab. The number enclosed in brackets shows the total number of records in the corresponding recipe.
Add	Adds records into the recipe according to the defined data type.
Delete	Deletes the edited content.

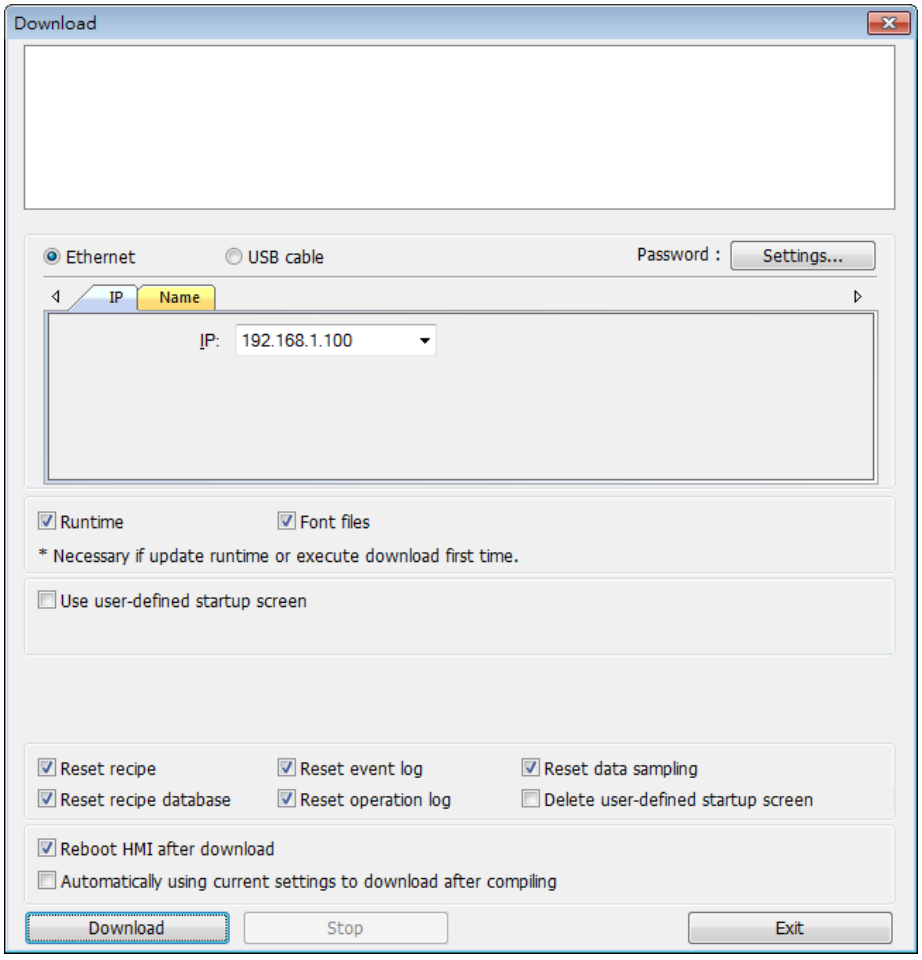
Example 1

1. Open [Recipe Records] dialog box to view all the recipe data built in [Definition] tab.
2. Click [Add] to add a new record and edit the content.
3. At the bottom of the dialog box shows the information of the selected item.
4. Click the up and down buttons to change the order of records.
5. Click [OK], a message pops up asking whether to apply the changes to database on PC. Clicking [Yes] will overwrite the old recipe data.



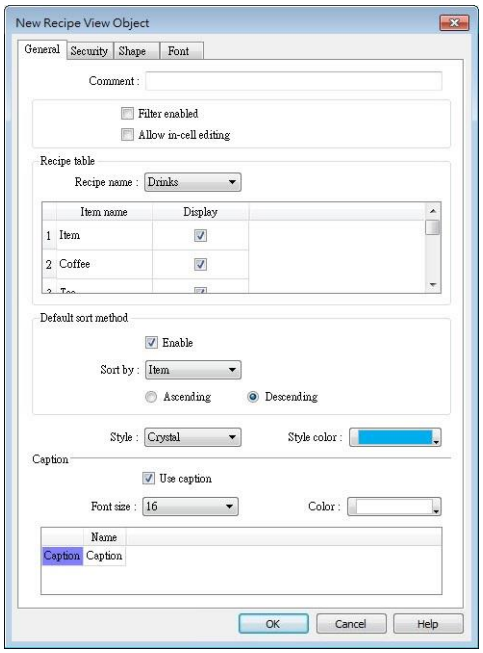
Note

- Each recipe can hold up to 10000 records.
- If click [Import], the current recipe records and also the recipes built in [System Parameter Settings] » [Recipes] tab will be overwritten by new data.
- The recipe records will be stored in *.exob file after compilation and will be downloaded to the HMI. These recipes are not allowed to be shared with other project files. If users need to modify the recipe content using Recipe Records and to download it to the HMI, make sure to select [Reset recipe database] check box. If not, the recipe database on HMI will not be updated.

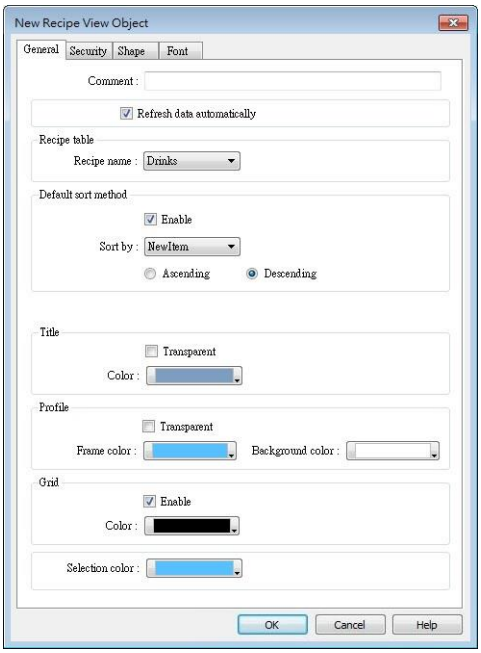


2.3. Recipe View

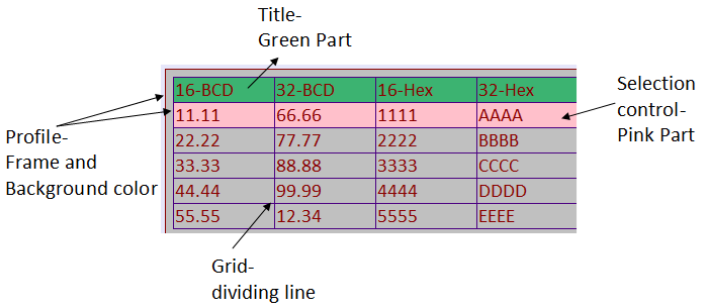
cMT / cMT X Series



eMT, iE, XE, mTV Series



The name of each part of the Recipe View object is shown in the following figure.



Setting	Description
Filter enabled (cMT / cMT X Series only)	When selected, entering keywords in Recipe View to search for specific text is possible.
Allow in-cell editing (cMT / cMT X Series only)	When selected, editing Recipe Database directly in Recipe View is possible.
Refresh data automatically	When selected, the system will automatically refresh Recipe View when recipe is changed; otherwise, Recipe View will be refreshed after window change.
Recipe table	**Recipe name**

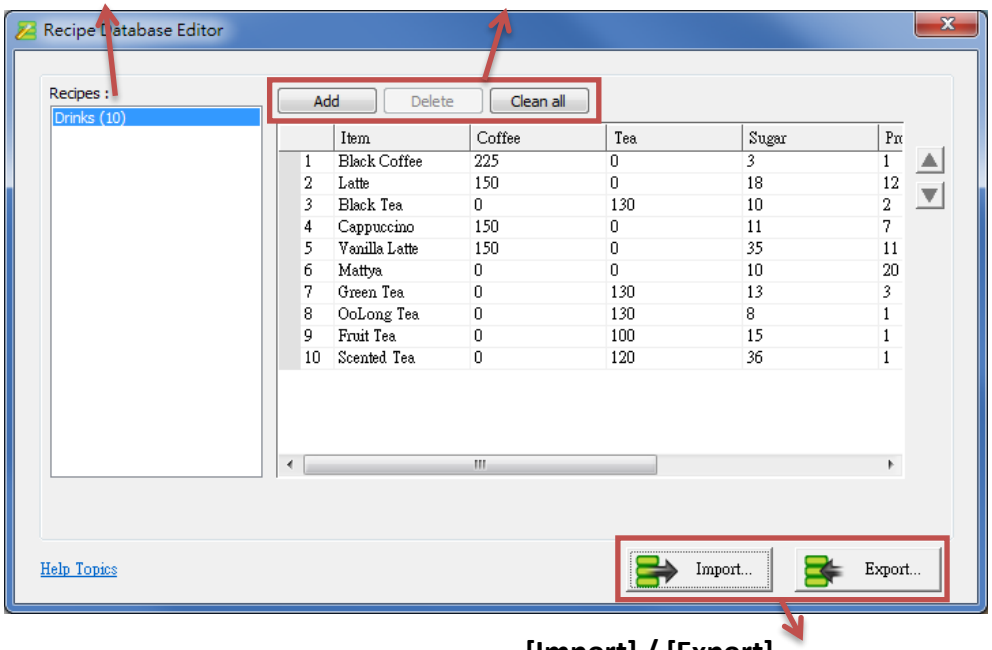
	Choose the recipe name or look for other recipes from the drop-down list. **Display** Choose an item to be displayed by selecting its checkbox.
Default sort method	Configure how the items are sorted. [Ascending] and [Descending] can be selected.
Style (cMT / cMT X Series only)	Available styles are: Default, Crystal, and Flat.
Caption (cMT / cMT X Series only)	With [Use caption] enabled, the text, font size, color, and name of the caption can be specified. (Use caption is only available when the selected style is Crystal or Flat.)
Title	The item name assigned in [Data/History] » [Recipe Database]. **Transparent** If selected, the title row has no shading; the color selection is not available.
Profile	The frame and background color of the object can be set. **Transparent** Select to hide the background, the color selection is not available.
Grid	The dividing lines between columns and rows. **Enable** Select to show the grid. **Auto fit short column** (cMT / cMT X Series Default style) The column width automatically adjusts to the size of the content.
Selection Control (non-cMT Series only)	Shading color of the selected row.

2.4. Recipe Database Editor

Use this tool to edit recipe data without opening EasyBuilder Pro, and then upload or download recipes by using Utility Manager.

[Recipe List]
Shows the recipes created in [System Parameter Settings]. The number enclosed in parentheses is the total number of records in one recipe.

[Add] / [Delete]
Click [Add] to insert a new item and edit.
Click [Delete] to delete the selected item.



[Import] / [Export]
Import *.db file for editing, and then export after editing.

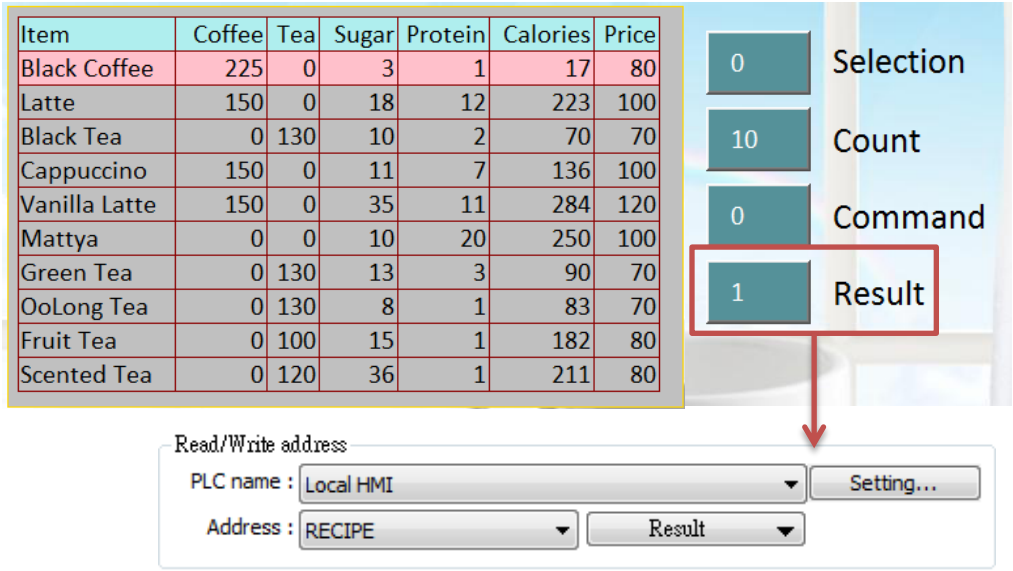
Example 2

1. Click [Recipe Database Editor] application to open the editing dialog box.
2. Click [Import] to import *.db files and edit recipe contents.
3. After editing, click [Export] to save the file to *.db format.
4. Download Recipe DataBase by using Utility Manager. When downloading, if select [Reset recipe database], the Recipe Database in HMI will be overwritten with the new settings.

3. Monitoring and Modifying Recipe Records

3.1. Monitoring Recipe Data

To watch / add / delete the displayed records, certain registers can be used. Create 4 Numeric Input objects, set addresses respectively to: RECIPE-Selection, RECIPE-Count, RECIPE-Command, and RECIPE-Result.

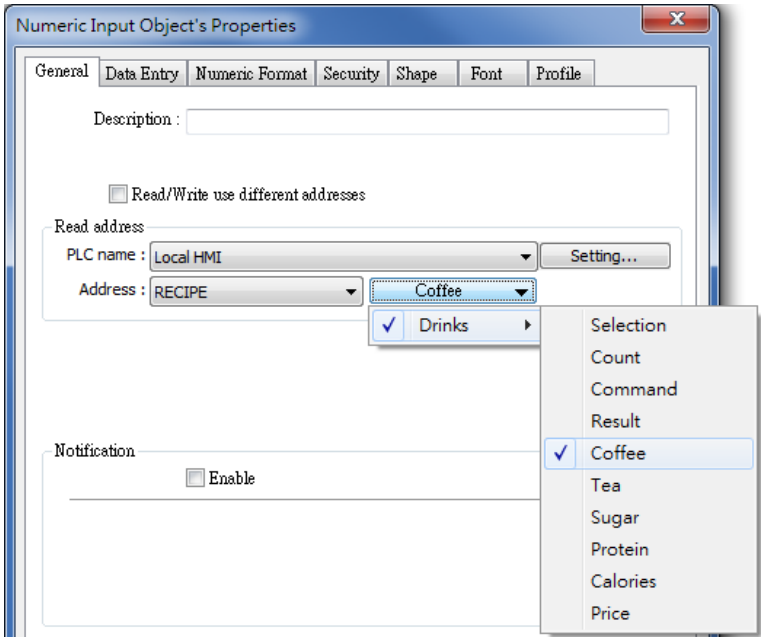


Setting	Description
Selection	The currently selected record. The records are numbered from zero. If choose the first record, the value of Selection will show “0”, and so on. When the value of [Selection] changes, the value in the relating register will change accordingly.
Count	The number of records in the recipe.
Command	Entering certain values will send certain commands to the selected record. Enter “1”: Add a new recipe record. Enter “2”: Update the selected recipe record. Enter “3”: Delete the selected recipe record. Enter “4”: Delete all recipe records. Enter “5”: Write the selected recipe record to PLC. Enter “6”: Update the recipe record selected from PLC.
Result	View the result of executing commands. Displays “1”: Command successfully executed. Displays “2”: The record does not exist. Displays “4”: Unknown command. Displays “8”: Records reach limit (10000 records), no new records can be added.

	Display “16”: Another command is being executed. Display “32”: Transfer command failed.

Recipe Name

The recipe data can be displayed if the item name in the recipe is selected in [Address].



3.2. Modifying Recipe Data on HMI

To modify recipe data, please create Numeric Input or ASCII Input objects first. Select the recipe item for address. After modifying, enter “2” in Command register to update.

Example 3

To use this demo project, please add a recipe in [System Parameter Settings] » [Recipes] tab first, and then edit the content in [Recipe Records].

Item name	Data type	Size	Display wi...	Decimal Pt.	Alignm...
Item	ASCII	10	12	0	Align left
Coffee	16-bit U...	1	7	0	Align right
Tea	16-bit U...	1	4	0	Align right
Sugar	16-bit U...	1	6	0	Align right
Protein	16-bit U...	1	7	0	Align right
Calories	16-bit U...	1	8	0	Align right
Price	16-bit U...	1	5	0	Align right

1. Create a [Numeric Input] object; select the item to modify in address field. Adjust the digits after the decimal point and set the upper and lower limit.

2. In Recipe View object, select the record to modify or enter the number of it in Selection field and then enter the new value to the corresponding register.

3. Enter “2” in Command register to update. Please note that entering “2” in Command will complete updating Recipe DataBase, and the setting in LB-9029 is irrelevant.

3.3. Transferring Recipe Data

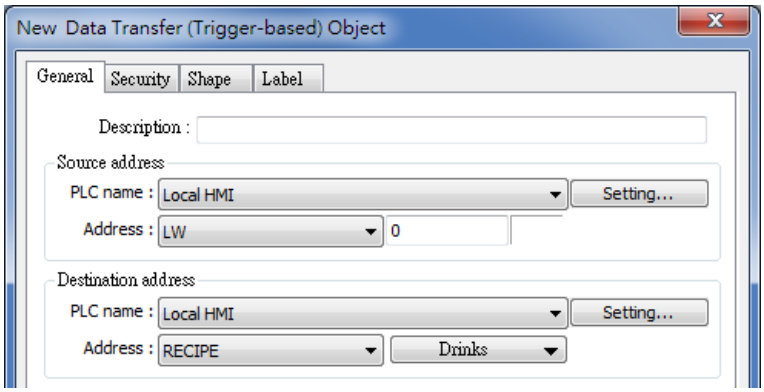
The edited recipe can be operated or adjusted using [Transfer (Trigger-Based) object or the designated register.

Example 4

This example explains how to transfer a complete recipe data. As shown in the following figure, the data to be transferred contains the following information: Item Coffee, Tea, Sugar, Protein, Calories.

Item	Coffee	Tea	Sugar	Protein	Calories	Price
Black Coffee	225	0	3	1	17	80
Latte	150	0	18	12	223	100
Black Tea	0	130	10	2	70	70
Cappuccino	150	0	11	7	136	100
Vanilla Latte	150	0	35	11	284	120

1. Create a Data Transfer (Trigger-based) object; designate the destination address to a specific recipe.



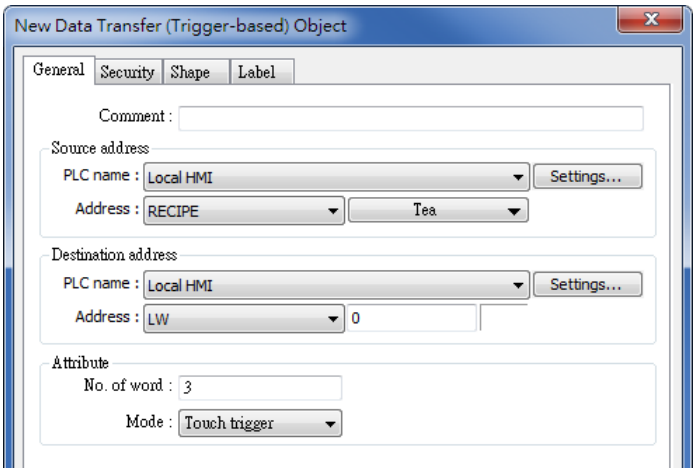
2. Create a local address object; the data format must be set identically to the recipe. For example, if a recipe includes two data types: 16-BCD and 32-BCD, the local address must set the same: LW-0 -> 16-BCD, LW-1->32-BCD.
3. In Recipe View object select the record to be transferred, or enter the number of the record in Selection.
4. Click Data Transfer (Trigger-based) object to transfer data. If transfer PLC data to Recipe register, enter “2” in Command register to finish updating.

Example 5

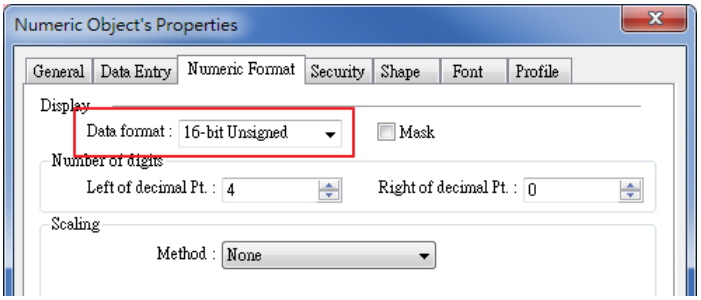
This example explains how to transfer recipe data of specified column. As shown in the following figure, when attempting to transfer data in Tea, Sugar, Protein columns, please follow the steps described.

Item	Coffee	Tea	Sugar	Protein	Calories	Price
Black Coffee	225	0	3	1	17	80
Latte	150	0	18	12	223	100
Black Tea	0	130	10	2	70	70
Cappuccino	150	0	11	7	136	100
Vanilla Latte	150	0	35	11	284	120

1. Create a Data Transfer (Trigger-based) object, set the source address to “Tea”, the destination address to LW-0, and number of words to 3.



2. Create three Numeric Input objects, set addresses respectively to LW-0, LW-1, LW-2. The value format must be identical to the settings in Recipe. As shown in the demo project, the format of Tea, Sugar, Protein items is 16-bit Unsigned, please set the same format in addresses LW-0, LW-1, LW-2.



3. In Recipe View object select the item to be transferred, or enter the number of the item in Selection.
4. Click Data Transfer (Trigger-based) object to transfer data. If transfer PLC data to Recipe register, enter “2” in Command register to finish updating.

3.4. Reading and Writing Bits in Recipe DataBase

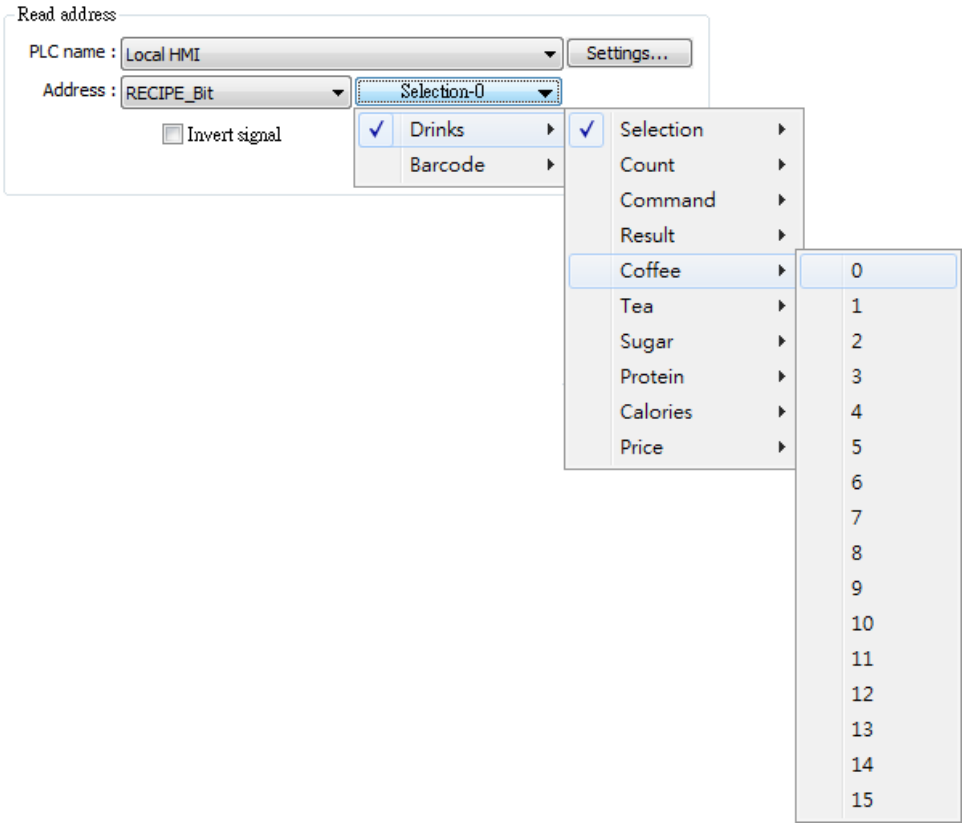
The bit address of recipe item can be read / written. This feature only supports Unsigned format.

Example 6

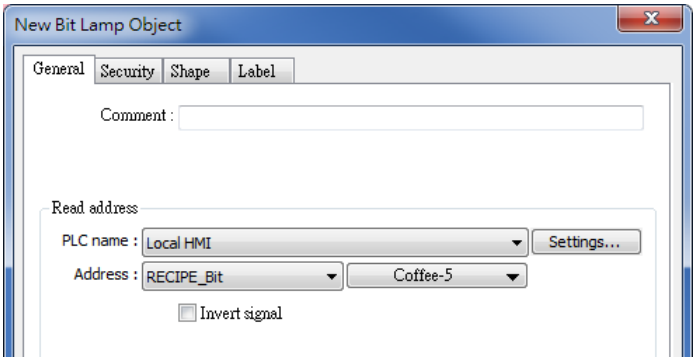
1. As shown in the following figure, except for “Item”, the data type of the rest items is 16-bit Unsigned.

Item name	Data type	Size	Display width	Decimal Pt.	Alignment
Item	ASCII	10	12	0	Left
Coffee	16-bit Unsigned	1	7	0	Right
Tea	16-bit Unsigned	1	4	0	Right
Sugar	16-bit Unsigned	1	6	0	Right
Protein	16-bit Unsigned	1	8	0	Right
Calories	16-bit Unsigned	1	9	0	Right
Price	16-bit Unsigned	1	6	0	Right

2. Create a Bit object, set the address to Recipe_Bit. When pointing to an item, its available number of bits will be displayed automatically. As shown in the following figure, the item “Coffee” can have 16 bits.



3. Select the read/write address. The address will be “Recipe_Bit*item name**bit number*”. As shown in the following figure, the 6th bit of Coffee is displayed in the address field.

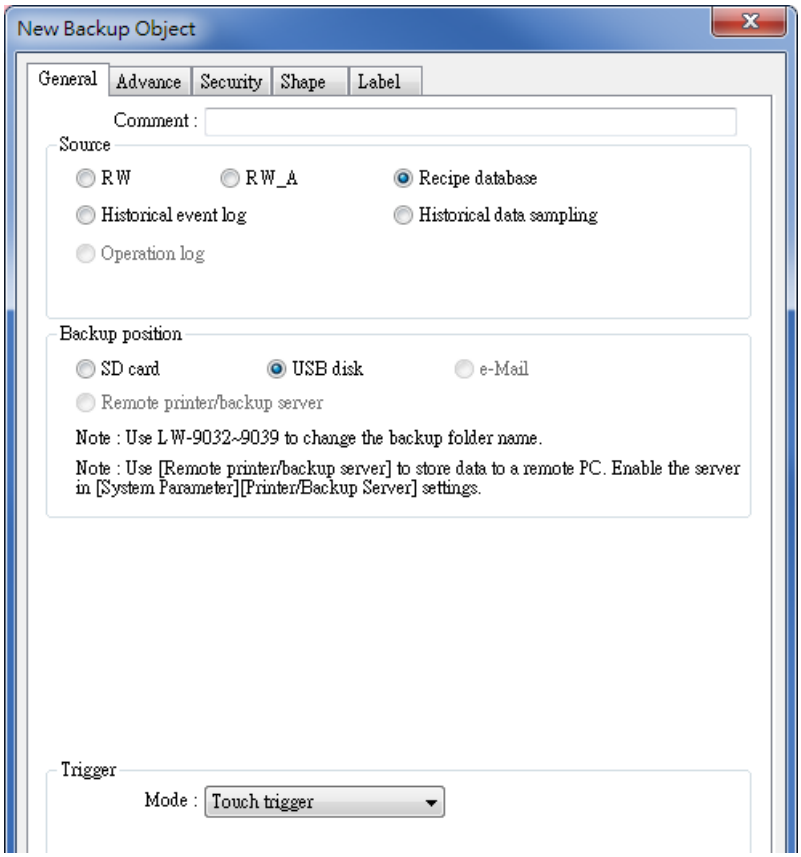


3.5. Backup Recipe DataBase

Backup object can be used to backup Recipe DataBase into USB drive / SD card, or send the data to the designated email box. The format of the backup data is .db.

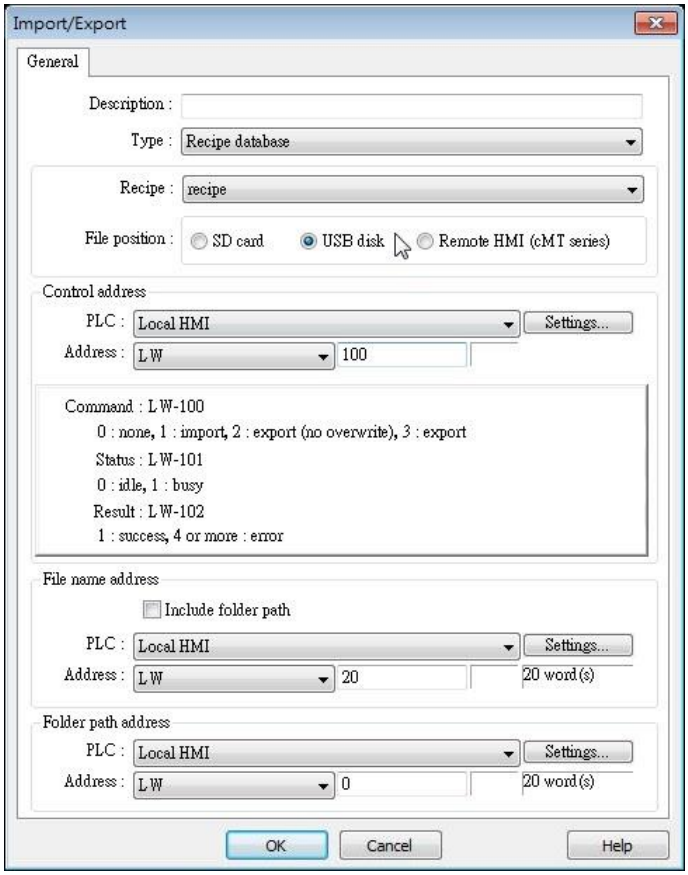
Example 7

Create a Backup object, select Recipe DataBase for source, and select the position to save the data.



3.6. Importing/Exporting Recipe Database

Open [Object] » [Import/Export] to import / export recipe database into USB disk, SD card, or cMT / cMT X Series HMI (importing recipe data is only supported on cMT /cMT X Series models).



Example 8

The following is an example on recipe export/import settings.

Field	Setting
File position	USB disk
Recipe	Recipe_A (or other recipe)
Control address	LW-100
File name address	LW-200
Folder path address	LW-250

1. Create two ASCII Input objects. Set address to LW-200 and LW-250 respectively.
2. Enter the file name in LW-200: 2015_recipe.csv.
3. Enter the folder path in LW-250: Setting.
4. Use a Set Word object to write value 3 to LW-100. Then, Recipe_A will be exported to the USB disk, in the “Setting/2015_recipe.csv” file.

Notes

- When performing “Export (no overwrite)” command, if the target file already exists, the export operation will be canceled, and the result value will be set to “4”. The following lists the result values and the information.

Result (HEX)	Information
0x1	Success.
0x4	File already existed and will not overwrite.
0x100	Data contains non-numeric data.
0x101	Path contains invalid string “..”.
0x102	Communication error while updating Recipe DB.
0x103	Error while reading Recipe DB information from project file.
0x200	General exception.
0x201	General status error.
0x202	Import to unknown database type.
0x203	Error while validating Recipe DB table definition.
0x204	Error while validating Recipe DB table data.
0x205	Error while writing Recipe DB table definition.
0x206	Error while writing Recipe DB table data.
0x300	File error: Unknown error.
0x301	File error: Empty file name.
0x302	File error: The external device does not exist.
0x303	File error: Invalid file name (directory or special files), or a folder with the same name already exits.
0x304	File error: Unable to remove file.
0x305	File error: Open file stream error.
0x306	File error: Unhandled BOM.
0x307	File error: Error while parsing CSV file (incorrect formats).
0x308	File error: Insufficient space on the external device.
0x400	Database general exception.
0x401	Database error: Unable to open table.
0x402	Database error: Unable to get rows.
0x403	Number of columns in CSV file and in Recipe DB do not match.

3.7. Searching Recipe Data by Macros

Recipe Query Functions enable searching a specific ID or data in a recipe.

Some functions are used to query recipe data:

1. RecipeGet Data: Get recipe data.
2. RecipeQuery: Query recipe data to obtain the number of records that meet the specified condition.
3. RecipeQueryGetData: From the result gained by RecipeQuery, get the data of the specific item.
4. RecipeQueryGetRecordID: From the result gained by RecipeQuery, get the specific record ID.
5. RecipeSetData: Write data into Recipe Database.

Name	RecipeGetData
Syntax	RecipeGetData (destination, recipe_address, record_ID)
Description	Get Recipe Data. The gained data will be stored in destination, and must be a variable. “recipe_address” consists of type name and item name: “recipetype_name.item_name”. “record_ID” specifies the ID number of the record in recipe being gained.

Example	macro_command main()
	int data=0
	char str[20]
	int recordID
	bool result
	
	recordID = 0
	result = RecipeGetData(data, "TypeA.item_weight", recordID)
	// From recipe "TypeA" get the data of the item “item_weight” in record 0.
	
	recordID = 1
	result = RecipeGetData(str[0], "TypeB.item_name", recordID)
	// From recipe "TypeB" get the data of the item “item_name” in record 1.
	
	end macro_command

Name	RecipeQuery
Syntax	RecipeQuery (SQL command, destination)
Description	Use SQL statement to query recipe data. The number of records of query result will be stored in the destination. This must be a variable. SQL command can be static string or char array. Example: RecipeQuery(“SELECT * FROM TypeA”, destination) or RecipeQuery(sql[0], destination) SQL statement must start with ”SELECT * FROM” followed by type name and query condition.

Example	macro_command main()
	
	int total_row=0
	char sql[100]="SELECT * FROM TypeB"
	bool result
	
	result = RecipeQuery("SELECT * FROM TypeA", total_row)
	// Query Recipe "TypeA". Store the number of records of query result in total_row.
	
	result = RecipeQuery(sql[0], total_row)
	// Query Recipe "TypeB". Store the number of records of query result in total_row.
	
	end macro_command

Name	RecipeQueryGetData
Syntax	RecipeQueryGetData (destination, recipe_address, result_row_no)
Description	Get the data in the query result obtained by RecipeQuery. This function must be called after calling RecipeQuery, and specify the same type name in recipe_address as RecipeQuery. result_row_no specifies the sequence row number in query result.
Example	macro_command main() int data=0 int total_row=0 int row_number=0

	bool result_query
	bool result_data
	
	result_query = RecipeQuery("SELECT * FROM TypeA", total_row)
	// Query Recipe "TypeA". Store the number of records of query result in total_row.
	
	if (result_query) then
	  for row_number=0 to total_row-1
	  result_data = RecipeQueryGetData(data, "TypeA.item_weight",
	  row_number)
	  next row_number
	end if
	
	end macro_command

Name	RecipeQueryGetRecordID
Syntax	RecipeQueryGetRecordID (destination, result_row_no)
Description	Get the record ID numbers of those records gained by RecipeQuery. This function must be called after calling RecipeQuery. result_row_no specifies the sequence row number in query result, and write the obtained record ID to destination.

Example	macro_command main()
	
	int recordID=0
	int total_row=0
	int row_number=0
	bool result_query
	bool result_id
	
	result_query = RecipeQuery("SELECT * FROM TypeA", total_row)
	// Query Recipe "TypeA". Store the number of records of query result in total_row.
	
	if (result_query) then
	  for row_number=0 to total_row-1
	  result_id = RecipeQueryGetRecordID(recordID, row_number)

	next row_number end if end macro_command

Name	RecipeSetData
Syntax	RecipeSetData(*source*, *recipe address*, *record_ID*)
Description	Write data to recipe. If success, returns true, else, returns false. *recipe_address* consists of recipe name and item name: "recipe_name.item_name". *record_ID* specifies the ID number of the record in recipe being modified.

Example	macro_command main()
	
	int data=99
	char str[20]="abc"
	int recordID
	bool result
	
	recordID = 0
	result = RecipeSetData(data, "TypeA.item_weight", recordID)
	// set data to recipe "TypeA", where item name is "item_weight" and the record ID is 0.
	
	recordID = 1
	result = RecipeSetData(str[0], "TypeB.item_name", recordID)
	// set data to recipe "TypeB", where item name is "item_name" and the record ID is 1.
	
	end macro_command

4. References

- For details of [Recipes] setting tab please refer to EasyBuidler Pro User Manual Chapter 5.11 or click here: Chapter_05_System_Parameter_Settings
- For details of [Recipe Records] please refer to EasyBuidler Pro User Manual Chapter 24.3 or click here: Chapter_24_Recipe_Editor
- For details of [Recipe View] please refer to EasyBuidler Pro User Manual Chapter 13.33 or click here: Chapter_13_Objects
- For details of macro, please refer to EasyBuidler Pro User Manual Chapter 18.6.7 or click here: Chapter_18_Macro_Reference
- Demo Project of Transferring Recipe Data: Recipe Transferring
- Demo Project of Searching Recipe data by Macros: Macro Recipe
- Demo Project of Backing up Recipe data into USB drive: Backup_Recipe_Database_to_USB_Demo
- Example of how to export Database files to CSV format: FAQ_50_Export_Recipe_DataBase_to_CSV_file
- Demo Project of Import/Export Recipe: Recipe Import/Export File_Browser Demo