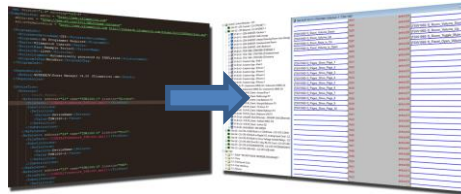


CONFIG-TO-CODE USER GUIDE

WHAT DOES CONFIG-TO-CODE DO?

Fundamentally, the tool takes information from one form – such as a spreadsheet, or another system – and generates SIMPL Windows programs from it.



These programs may be skeletons, on which to build a bigger system, or they could be complete, working, programs and Config-to-Code will then allow you to compile and upload the output straight to a Crestron processor.

The nature of the programs that Config-to-Code will generate is defined by "plugins" that are shipped with the tool. Ultamation will produce plugins for specific use-cases – such as building BACnet programs from an HVAC report or creating a DT8 lighting integration for Crestron Home from a lighting load schedule spreadsheet – but 3rd parties can create their own plugins too.

If you have a suggestion for plugin we'd love to hear from you.

WHO IS THE INTENDED AUDIENCE?

Config-to-Code is a tool that can be useful for both seasoned Crestron programmers, and for those without ANY programming skills or experience.

For the seasoned programmers

Config-to-Code can take simple input and generate the largest of programs that may have many hundreds of repeated devices and code blocks with tens of thousands of signals, and do this without typos, without the drudgery, in a few seconds. This offers a clear competitive advantage over programmers doing things the traditional way. Why spend a day defining duplicate BACnet objects when you can fill in a spreadsheet and have the program, complete with your own custom logic, in a matter of minutes and at the same time avoid the risk of missing a signal mis-type and a possible service call?

For the non-programmers

The use-cases will be different, but where the only solution to a problem is to get into custom programming, Config-to-Code offers a way to solve that problem without ever touching SIMPL Windows or being able to tell the difference between a "logic wave" and "jammed signal" – neither of which have anything to do with hand gestures or fruity preserves.

One such situation is control of DALI DT8 lighting within Crestron Home. Config-to-Code, in conjunction with our SIMPL Home integration suite, provides a solution to take a simple spreadsheet table of lighting loads, and – with only a few clicks – start adding colour tuneable (DALI DT8) lighting groups to a Crestron Home system.

BEFORE YOU START

Although you don't need to open any Crestron software to use Config-to-Code, the tool does depend on you having a number of pre-requisites installed on your machine. Since the tool makes use of Crestron's development tools, the software is only compatible with Windows systems.

-  **Config-to-Code**
You will need to install the tool using the software installer. This can be downloaded from <https://shop.ultamation.com/index.php/category/13-tools>
The installer will also install the current suite of program generator plugins.
-  **SIMPL Windows**
This provides the fundamental tools required to build a Crestron program.
-  **SIMPL+ Plugin**
Required for compilation of certain modules.
-  **Crestron Database**
-  **Device Database**
These provide information about Crestron products.

Some program generator plugins have additional requirements – such as SIMPL Home modules. In such cases, the specific user guide for that plugin will inform you of these requirements and where to obtain the software.

HOW DOES IT WORK?

Config-to-Code has been designed for simplicity and takes a "wizard" approach.

The wizard will guide you through the following sequence of steps:

-  **Initialisation**
This step performs the checks and balances to ensure Config-to-Code can continue (such as checking that Crestron databases are installed).
-  **Select a program generator plugin**
You'll be offered a list of available program generators, each with a brief summary of what its use-case would be.
-  **Settings**
Each program generator will have one or more options that will need to be configured. This is simple information, such as the IP address and password of a Crestron Home system, or the file path to a config file. The plugin-specific documentation will explain each of these in more detail.
-  **Build**
This step allows you to confirm you're ready to build a system. Each program generation will consume one or more "credits" depending upon the plugin being used (some plugins can offer free generation, or demo modes where a credit is not consumed but the output will be limited).
-  **Upload**
If the program generator creates a finished program, you'll be offered the chance to compile and upload the program to a Crestron processor.
-  **Completion**
If all has gone well, the final page will give you any "next steps" such as moving on to the configuration of Crestron Home.

Please refer to the specific plugin you wish to use for details information.

PLUGIN SPECIFIC GUIDES

Config-to-Code is currently shipped with the following program generators:

-  DIN-DALI-2 to SIMPL Home Builder
-  Crestron Home to Custom Integration Builder
-  BACnet Device Tree Builder¹

Full details for each plugin are given in the next section.

DIN-DALI-2 to SIMPL Home Builder

Takes a simple CSV format, detailing the DIN-DALI-2 devices and either DT8 (tuneable white DALI ballasts) or DT6 (standard dimmable white) and generates a SIMPL Home program that presents Crestron Home with each of the lighting groups configured as an appropriate native lighting load.

To use this program, Crestron Home will ALSO require a SIMPL Home licence for each project.

Crestron Home to Custom Integration Builder

Crestron Home can now be used as a lighting solution alongside custom projects, thereby replacement legacy D3Pro solutions. As each of the room, scene and load modules has a unique integration ID within Crestron Home, this plugin is used to query a live Crestron Home system, and build a skeleton SIMPL Windows program containing the integration modules, along with their pre-populated IDs.

All that is required is for those modules to be connected up to new custom programming, or copied into an existing program and connected to that logic.

BACnet Device Tree Builder

Taking a simple CSV format, detailing one or more BACnet hosts and objects, this plugin will create a program containing the BACnet object tree – which can be extremely time consuming to build – and can also include connected logic.

The signal connections and logic does require additional XML definitions which can be created by hand (refer to the SIMPLified 2 documentation) or extracted from a "template" program using the SIMPLified 2 Productivity Entitlement.

When combined with XML snippets, a complete – fully functional – program can be generated using your own custom logic.

Note: depending on the number of BACnet objects defined, you may also need a BACnet licence for the Crestron custom processor.

¹ The BACnet Device Tree Builder was originally developed against SIMPLified2 which provides the core technology for Config-to-Code. To get the most from this program generator, we suggest programmers consider the SIMPLified2 Productivity Entitlement which enables export of existing program logic and devices.

DIN-DALI-2 TO SIMPL HOME BUILDER

After selecting the DIN-DALI-2 to SIMPL Home Builder (for use with Crestron Home) you will need to complete the plugin settings.

Setting	Description
Output Folder	This is the folder where your generated program will be stored. This usually also contains various intermediate files, so we recommend using a new folder for each program.
CSV File Path	This is the configuration file used to define the lighting groups and DALI hardware. See below for details of the format.
Processor Type	Here, you will select the processor model you wish to build the program for. For example, if you have a CP4 custom processor, select "CP4" from the drop-down.
SIMPL Home Host Port	This is the port you wish to run SIMPL Home on – unless you have a specific need to do otherwise, leave this at the default of 33333
SIMPL Home Zip File	The program generator REQUIRES SIMPL Home. Please download the SIMPL Home Example file from https://shop.ultamation.com/index.php/product/159-simpl-home-driver-for-crestron-home The plugin will automatically extract the necessary files from the ZIP archive.

Only when all of these settings have been completed, will you be able to move onto the build stage.

DIN-DALI-2 Config File Format

The plugin requires a file with a specific structure and saved in the CSV format (all spreadsheet software supports CSV – "Comma Separated Values").

	A	B	C	D	E	F	G	H	
1	Friendly Name	DALI Device IP ID	DALI Loop Number	CCT Recall	Fade Rate	DALI Group Number	Minimum CCT	Maximum CCT	
2	DIN-DALI-2-ABC	4							
3			1	1	300				
4	ABC Dual Channel 1					1	1800	8000	
5	ABC Dual Channel 2					2	2000	6000	
6			2	2	200				
7	ABC Dual Channel 3					1	1876	8765	
8	ABC Single Channel 1					2			
9	DIN-DALI-2-DEF	5							
10			1	1	400				
11	DEF Single Channel 1					1			
12	DEF Dual Channel 1					2	2200	3000	
13			2	2	100				
14	DEF Single Channel 2					1			
15	DEF Single Channel 3					2			
16									

Each line of the file has a specific meaning.

Top Line

This declares the meaning of each column, and **MUST** be spelled exactly as shown. While the order of columns can be modified, we strongly suggest you use the format as shown.

Heading	Description
Friendly Name	An identifier used to name the objects within the generated program
DALI Device IP ID	The IPID of the DIN-DALI-2 device (configured on the device itself)
DALI Loop Number	The DALI loop number (1 or 2) for the hardware device defined above
CCT Recall	1 = Direct Analog, 2 = Scene Approximation (See Crestron documentation)
Fade Rate	The time in Milliseconds for fades of the groups listed below

DALI Group Number	A unique DALI group number – as configured during DIN-DALI-2 commissioning
Minimum CCT	The minimum colour temp (warm) for the group (in Kelvin)
Maximum CCT	The maximum colour temp (cool) for the group (in Kelvin)

Subsequent lines in the file are:

DIN-DALI-2 Device

Define a piece of physical hardware. The only columns relevant here are "Friendly Name" which is used as a label, and "DALI Device IP ID" which defines the position for the device in the processor's IP Table.

This must then be followed by:

DIN-DALI-2 Loop Information

Applicable columns are "DALI Loop Number" which will be 1 or 2 for the DIN-DALI-2 physical bus connection, "CCT Recall" which is a value of 1 or 2 (see Crestron module documentation for meaning) and "Fade Rate" which is the timing in ms for fades on that loop.

This must then be followed by one or more:

DALI Group Definition

Each unique DALI group on the loop is then listed, one group per line. The applicable fields are "Friendly Name" which is used to present the lighting load in Crestron Home and "DALI Group Number" which will match the group defined during commissioning.

If both "Minimum CCT" and "Maximum CCT" are specified, then the system will generate a tuneable white lighting device (with the specified values) and DALI DT8 will be used. If these values are omitted, then the light is defined to be single channel (DT6) both in the program and in Crestron Home.

Build and Upload

If demo mode is selected, the program generation will be limited to 4 DALI groups.

Once you have entered a valid Config-to-Code key with sufficient credit available, press Generate to build the program. You will then be offered an opportunity to upload the program to your processor.

DIN-DALI-2 Configuration

Please note – the DIN-DALI-2 must be commissioned using the standard Crestron tools to assign DALI ballasts to groups.

Finishing Up

Once the program is up and running, please refer to the SIMPL Home documentation to load the driver onto Crestron Home and add the lights made available to the system.

BACNET DEVICE TREE BUILDER

After selecting the BACnet Device Tree Builder you will need to complete the plugin settings.

Please Note: This assumes you have already completed your lighting commissioning in Crestron Home.

Setting	Description
Output Folder	This is the folder where your generated program will be stored. This usually also contains various intermediate files, so we recommend using a new folder for each program.
CSV File Path	This is the configuration file used to define the BACnet hardware and associated logic. See below for details of the format.
Processor Type	Here, you will select the processor model you wish to build the program for. For example, if you have a CP4 custom processor, select "CP4" from the drop-down.

BACnet Config File Format

A CSV file is required to define the BACnet structure and devices. The image below shows our example document **BACnet Input Example.csv** downloadable from the SIMPLified 2 shop page.

This is a basic example, for a more detailed use case, refer to the **SIMPLified 2 Bacnet Worked Example** document, downloadable on the shop page or accessible by the SIMPLified 2 help menu.

Omit	DataType	Name	Id	Instance	PortNumber	Covinc	Sign	Polarity	States	CovSub	Scan	Priority	CovLife	DeviceManifest	LogicManifest	Units	Decimals	Substitutions
	Hosted		101															
	AI	Room 123 Temp	1001												BACnet_Analog_Input.xml	53	1	
y	AV	Room 123 SetPoint	1002												RoomMode.xml	47		
	BI	Room 123 Demand	1003															
y	BV	Room 123 Mode	1004												BACnet_Binary_Value.xml			
y	MI	Room 456 Wobble	1005												BACnet_Multistate_Input.xml			
y	MV	Room 456 Wobble	1006															
	Remote		301															
	BI																	
	Remote		201															
	AI	Remote Analog Input	2001															

When this file is passed to the BACnet Device Tree Builder, it will output an smw file with the following BACnet structure. And you can see all the fields have been populated with the information provided in the csv file. This is the Hosted Analog Input Object from row three of the example file:

<u>Value_Hi_Word#</u>	PresentValueHiWord
<u>Value_Lo_Word#</u>	PresentValueLoWord
<u>Status_Flags#</u>	StatusFlags
<u>Event_State#</u>	EventState
<u>Units#</u>	Units
ObjectID	1001d
ObjectName	Room 123 Temp
Units	pascals
COV Increment	1d
Signed	No
DecimalPlace	1

Creating the BACnet csv File

You can use the example document as a template for your own BACnet devices. Simply fill in the device information in each row. If you are using a spreadsheet like the example, the document should be saved as a csv file before using in Config-to-Code.

To add *hosted devices*, create one row with 'Hosted' as the DataType, then list all the hosted devices under it, using one row per device. **Note:** There should only be one Hosted section in the file.

To add *remote hosted devices*, add a row with 'Remote' as the DataType, then list the remote devices underneath using one row for each device. You can have multiple remote host sections.

The XML for Device and Logic Manifests can be created by hand or using SIMPLified 2 with the productivity entitlement. This is described in the SIMPLified 2 documentation (<https://shop.ultamation.com/index.php/hikashop-category-information-menu-129/product/100-simplified-2-productivity>)

Columns Explained

Listed below are the column headers for the BACnet Transformer input file with a description for each

Column Header	Description	Value Type
Omit	Type anything into this column if you want to omit the device in this row from being included in the generated program. If the field is empty then the device will be included. Note: if a Hosted or Remote row is labelled to be omitted, all sub devices will also be omitted.	Any
DataType	To begin a Hosted Devices section, write 'Hosted' in this field, then the rows beneath will be the sub devices. To start a Remote Hosted Devices section, write 'Remote' here with remote devices in the rows beneath. For a device row, this should be the device's abbreviation, e.g. a Analog Input Object would be 'AI' in this field. A Multistate Output Object would be 'MO'.	String
Name	The object name parameter in the symbol.	String
Id	The symbol's object ID parameter value.	Int
Instance	Defines the value that will be substituted for "bacnetDevice" in the signal and logic definitions	String
PortNumber	The symbol's port parameter value.	Int
Covinc	The symbol's COV Increment parameter value.	Int
Sign	Whether the device is signed or unsigned.	Yes or no
Polarity	The symbol's polarity parameter value.	
States	The symbol's Number of States parameter	Int
CovSub	The symbol's COV Subscribe parameter	Yes or no
Scan	The symbol's Scan Rate parameter in seconds	Int
Priority	The symbol's Priority parameter	Int
CovLife	The symbol's COV Lifetime parameter	Int
DeviceManifest	Specifies that an external Manifest file is to be loaded and used to define the signal connections for the hardware object and any default parameter values. These values can then be overridden from the CSV file if they are defined (e.g. "Decimals" in this example.)	File name
LogicManifest	Specifies a single logic folder.	File name
Units	The symbol's Units parameter	Int
Decimals	The symbol's DecimalPlace parameter	Int
Substitutions	Provides a way for the CSV file to override signal, comment and parameter values with row specific values. Name value substitution pairs are defined by NAME=VALUE and multiple values are separated using the pipe ' ' symbol	String

Build and Upload

This program generator does not offer a demo mode.

Once you have entered a valid Config-to-Code key with sufficient credit available, press Generate to build the program. You will then be offered an opportunity to upload the program to your processor.

Finishing Up

Once the program is up and running, you can use a program such as YABE (Yet Another BACnet Explorer) to verify the BACnet objects are advertised as expected and then connect up to your chosen control (e.g. Crestron Home).

SUPPORT

If you require support, please contact Ultamation at support@ultamation.com

Please provide a full description of your issue, including any configuration files you're using.