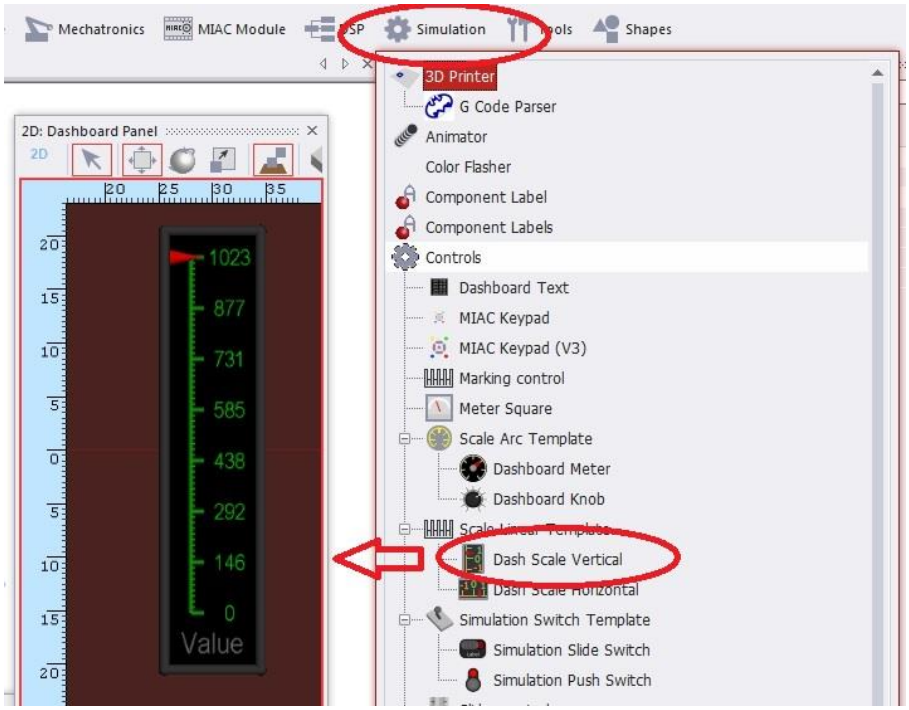
Example of creating a Flowcode 7 component for an I²C based device, such as a LM75B digital temperature sensor.

Start by creating a new Flowcode project. It does not matter what the target is, although it helps if you choose a target that supports I²C. We can now start our new component by using existing base components.

Building from components

The main function the device, such as the LM75B used in this example, is to provide us with a value for the measured temperature.
So we need a means to set this value when we are running the simulation of an application that uses our new component.

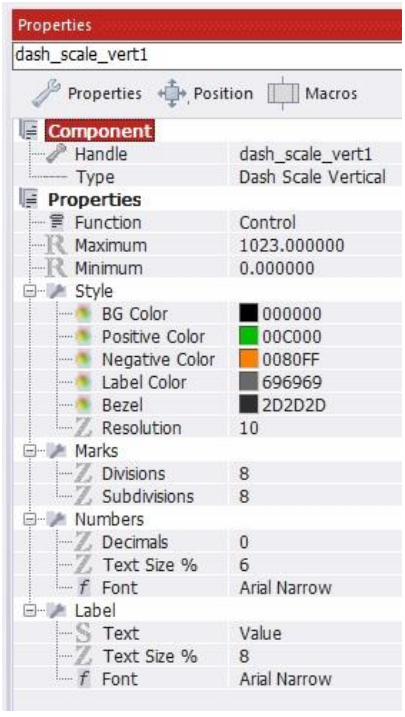
A linear scale is a good way to provide it, so let's add one.
Select the "Dash Scale Vertical" from the "Simulation" list and drag it onto the panel.



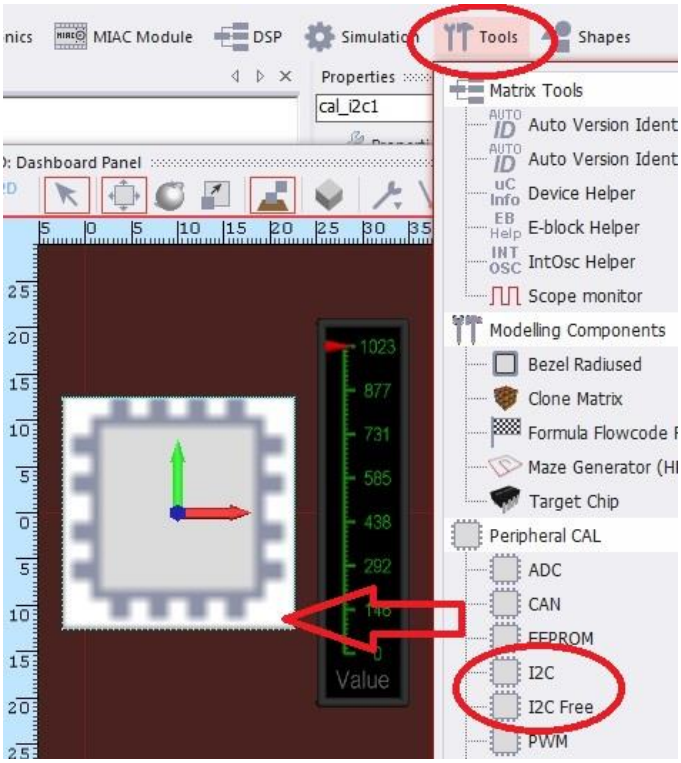
Set the scale component properties,
such as the minimum, maximum and resolution values.

For simplicity, for now we have set the minimum to 0
and the maximum to 1023, the range for a 10 bit resolution.

For advanced users:
This range could be set to floating point values
to indicate an actual temperature value.



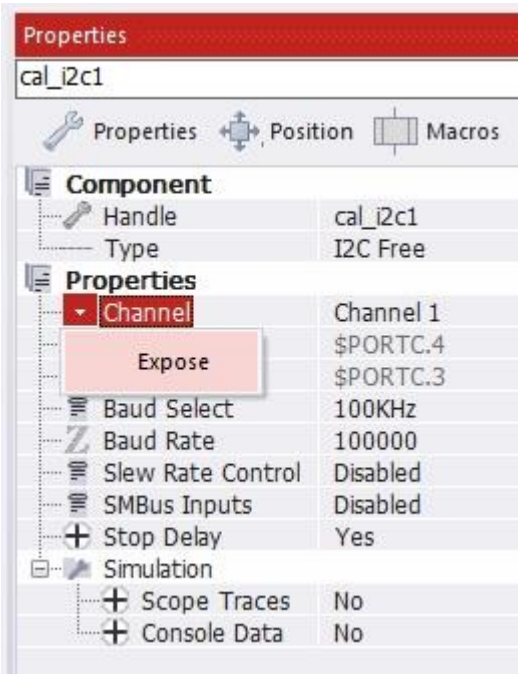
The LM75B device interface is I²C, so the Flowcode I²C CAL component will help us, drag one to the panel from Tools component list.



Set the I²C properties, such as the Baud Rate, to match the device.

The properties that are going to be user configurable in our new component need to be set to “Expose”, so do this for:

- Channel
- SDA
- SCL

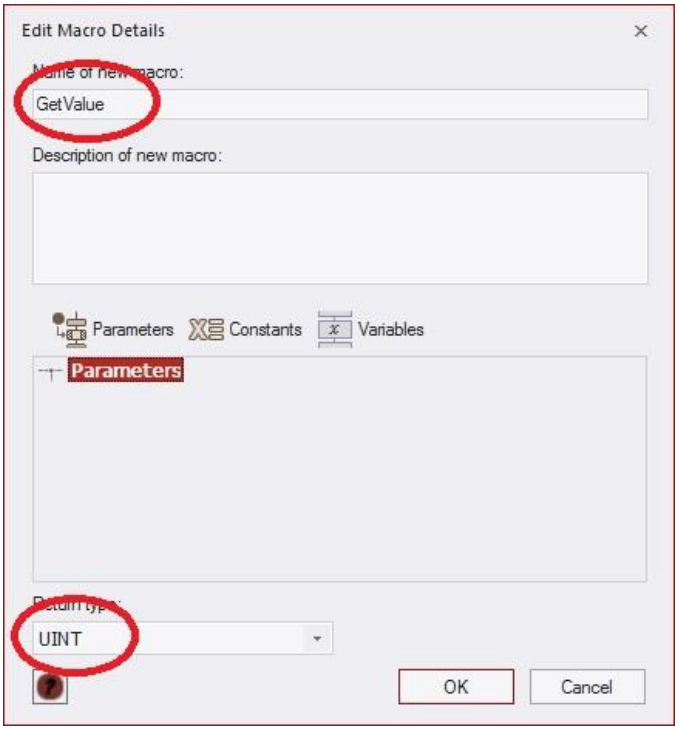


Creating the component interface

From the device datasheet decide the API (macros) that you require to access the device functionality.
For this example we are assuming a device that provides a digital value of temperature, but it could just as easily be a voltage, current or some other sensor.

Create a new macro, called say GetValue, to do this go to menu MACRO->New

Enter the name of the macro, “GetValue”.



Set the Return type to UINT.

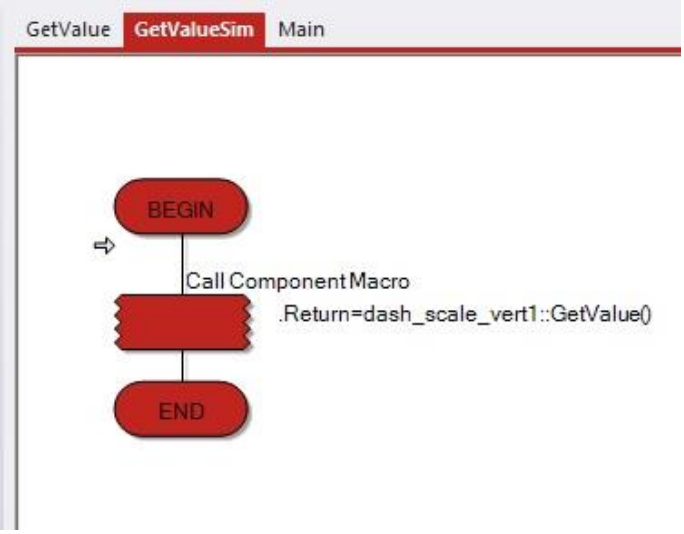
This macro “GetValue” will be the macro that is part of the API we provide for the downloaded target component.

We also require a macro that provides the same functionality for the simulation.

So click on the GetValue tab, then go to menu MACRO->Duplicate,
enter “GetValueSim” as the name of this new macro and click OK.

In GetValueSim macro add a call to
dash_scale_vert1::GetValue
and put the return value into “.Return”

This will provide the slider value when our new component is run in simulation.



Creating additional component properties

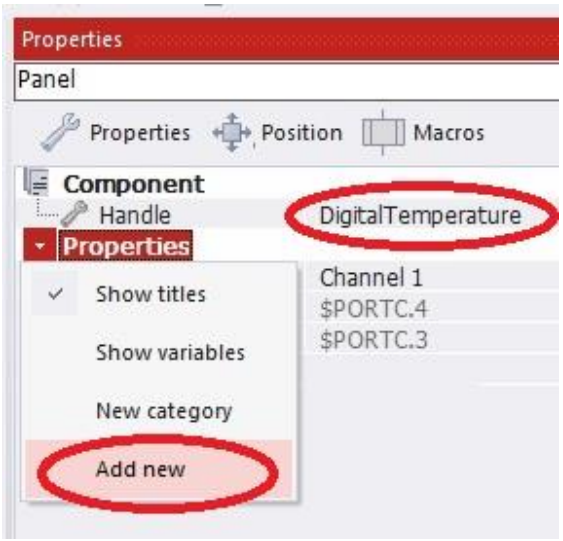
In the Properties window, select the “panel”

Give our new component a name, such as “DigitalTemperture”

Note that we can see our “exposed” properties, Channel, SDA and SCL from the I²C CAL.

The LM75B I²C device has a bus address that is configurable via the A0, A1 and A2 pins.

So here we use “Add new” property



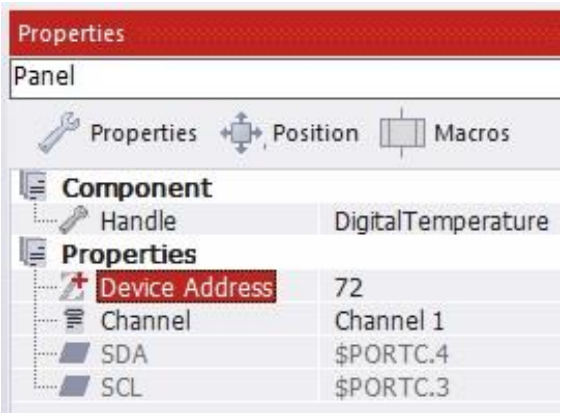
Give this new property the cosmetic name “Device Address”

Select “Unsigned integer” as the property type.

We also give the property a variable name of “DEVICE_ADDRESS”. This is the name that will be used to reference this value within the component Flowchart code.

Now we see the properties that will be available in our component.

These user configurable properties can be given default values. For example a “Device Address” of 72 can be set, for the case where address pins A0, A1 and A2 are all pulled low.
Enter this value into the box alongside the property name.

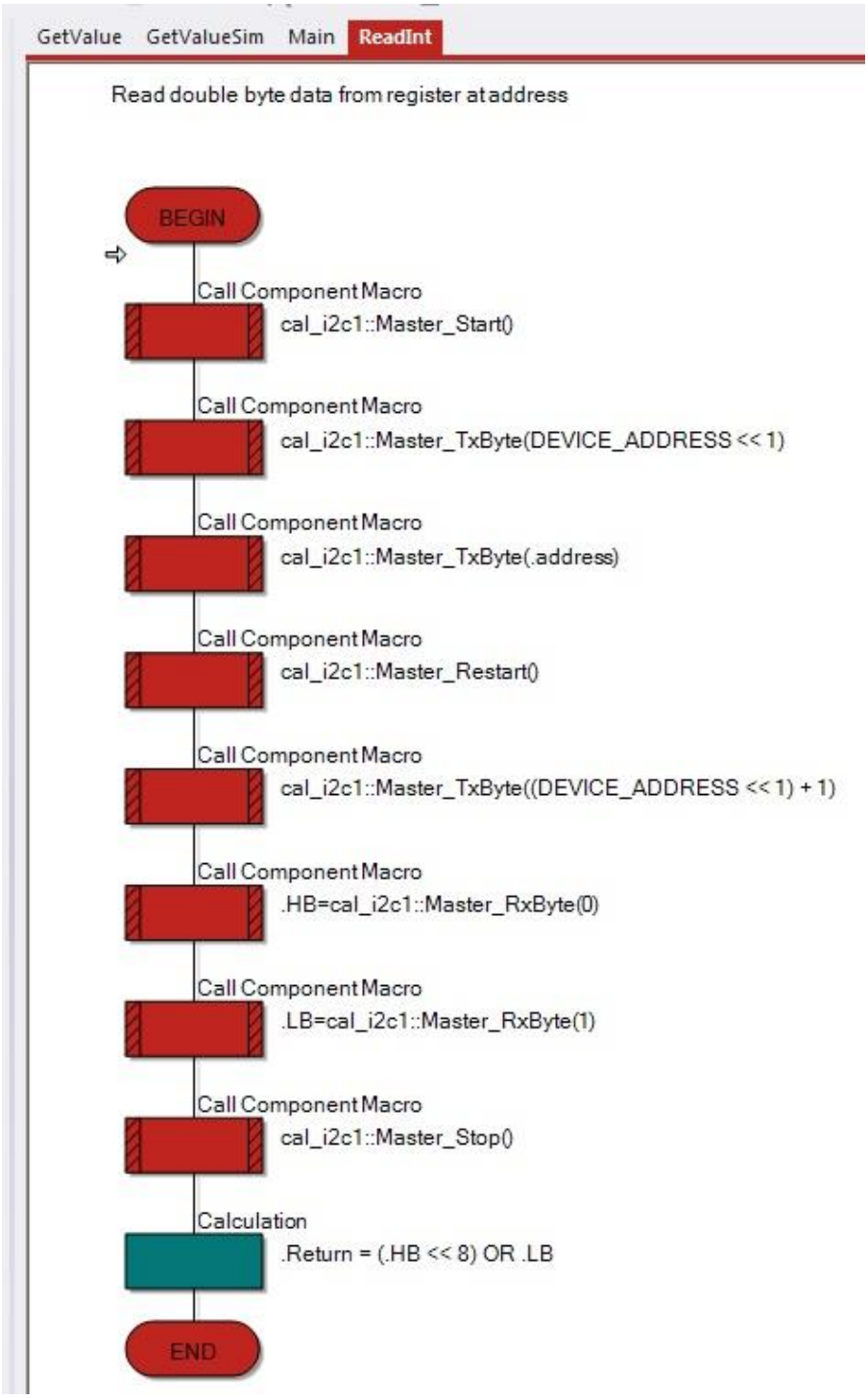


Completing the component macros

Getting the value from the actual device involves reading a pair of register values from the device, so it makes sense for us to first write a generic macro that reads a pair of bytes from a given address in the device and returning it as an integer.

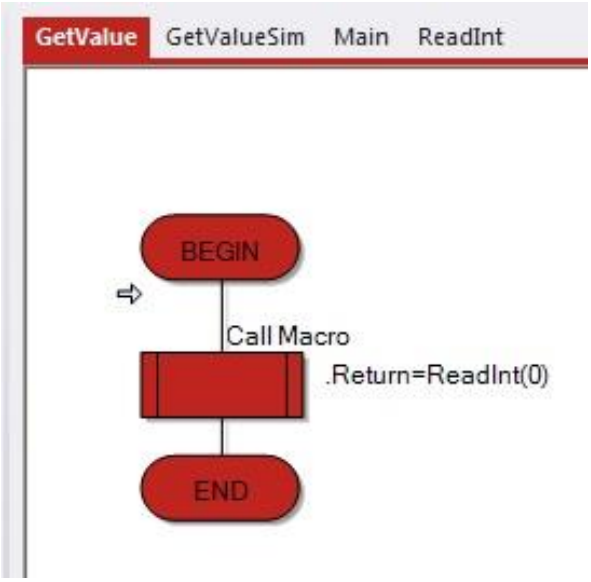
So here is a new macro called ReadInt that has a parameter "Address" and returns a UINT.

The devices' I^2C bus address is determined from our property DEVICE_ADDRESS

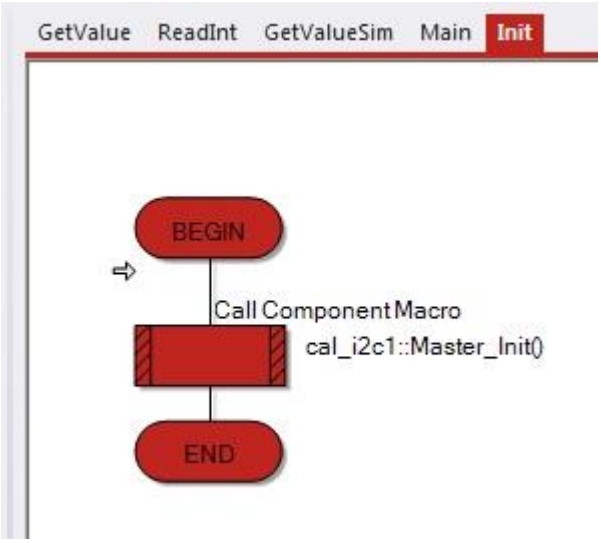


For this particular device the digital temperature is read from the first two locations of its internal registers.

So our GetValue macro simply calls ReadInt with an address parameter of 0.



We also require an initialisation macro that initialises the I^2C CAL component.



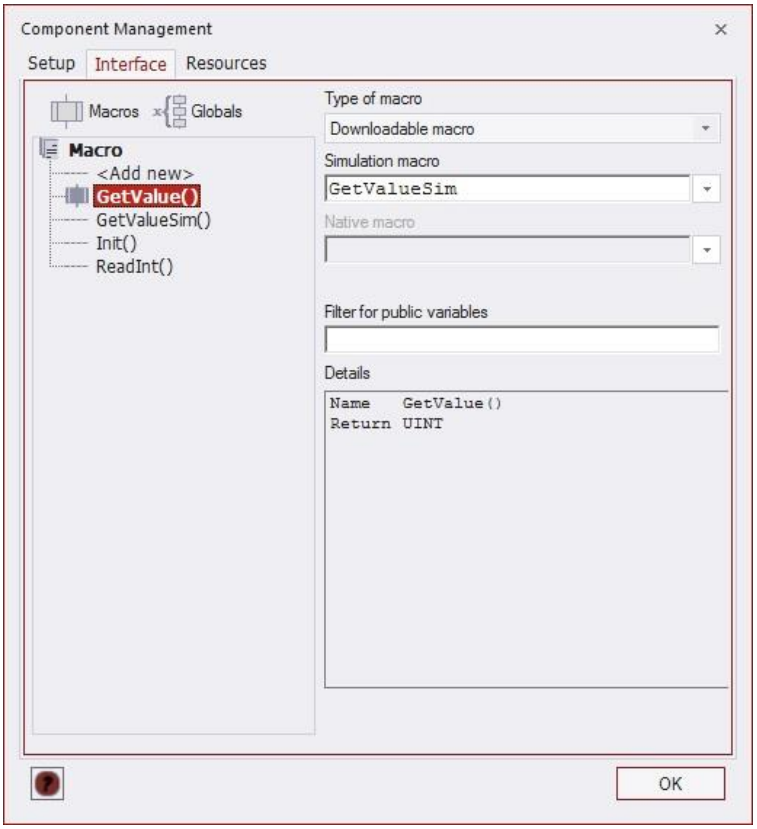
Exporting the component

We are now ready to make our component.

Get the configuration dialog from menu FILE->Component configuration

Go to the “Interface” tab and set:

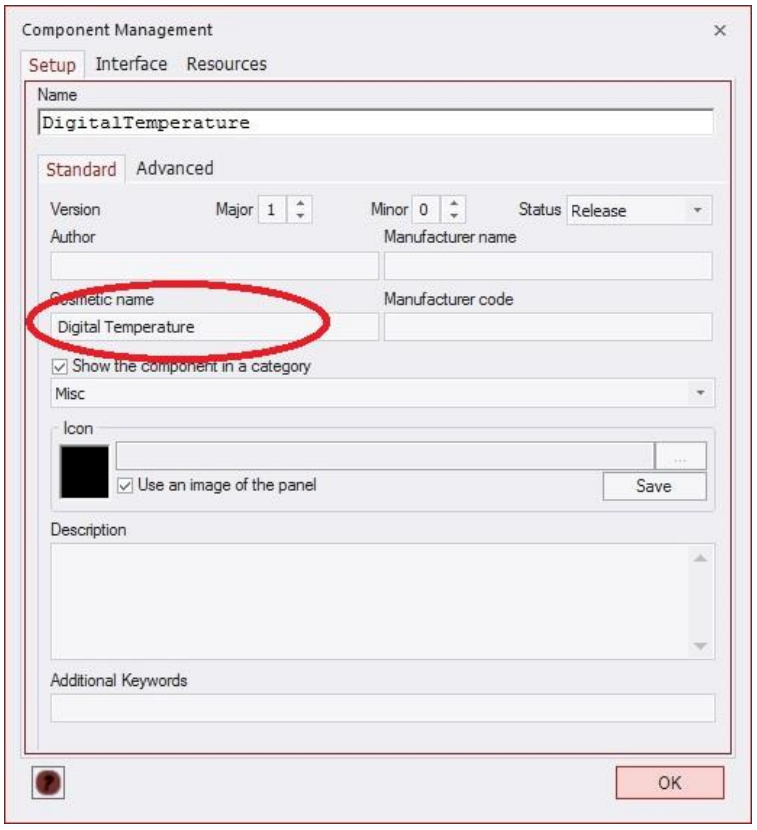
- “GetValue” macro as type “Downloadable macro” and set its Simulation macro as “GetValueSim”
- “GetValueSim” as type “Hidden macro”
- “Init” macro as type “Downloadable macro”
- “ReadInt” macro as type “Downloadable macro”



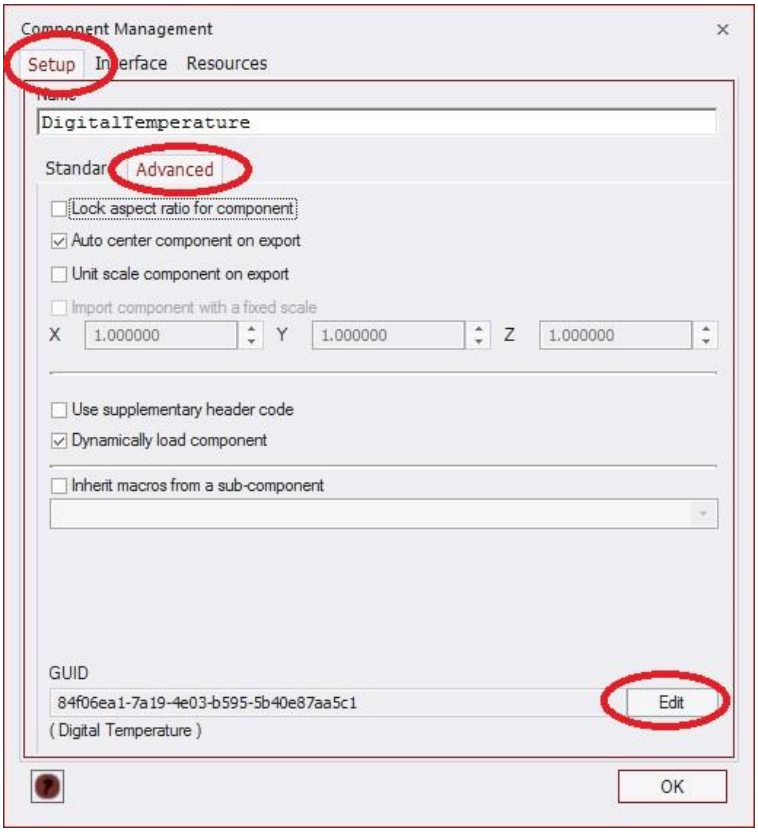
Go to the “Setup” “Standard” tab and give the component a Cosmetic name.

The category that the component will appear under can be set. “Misc” is the default.

Other details can be added, such as a description of the component and keywords for the search feature of Flowcode.



Go to the ”Setup” “Advanced” tab



Click the GUID Edit button.
Click the “New” button and then “OK” to create a new GUID.

NOTE that this only requires doing once and will be saved with the project. The GUID identifies the component in Flowcode. Only change this again if you are creating an entirely new component based on this one.

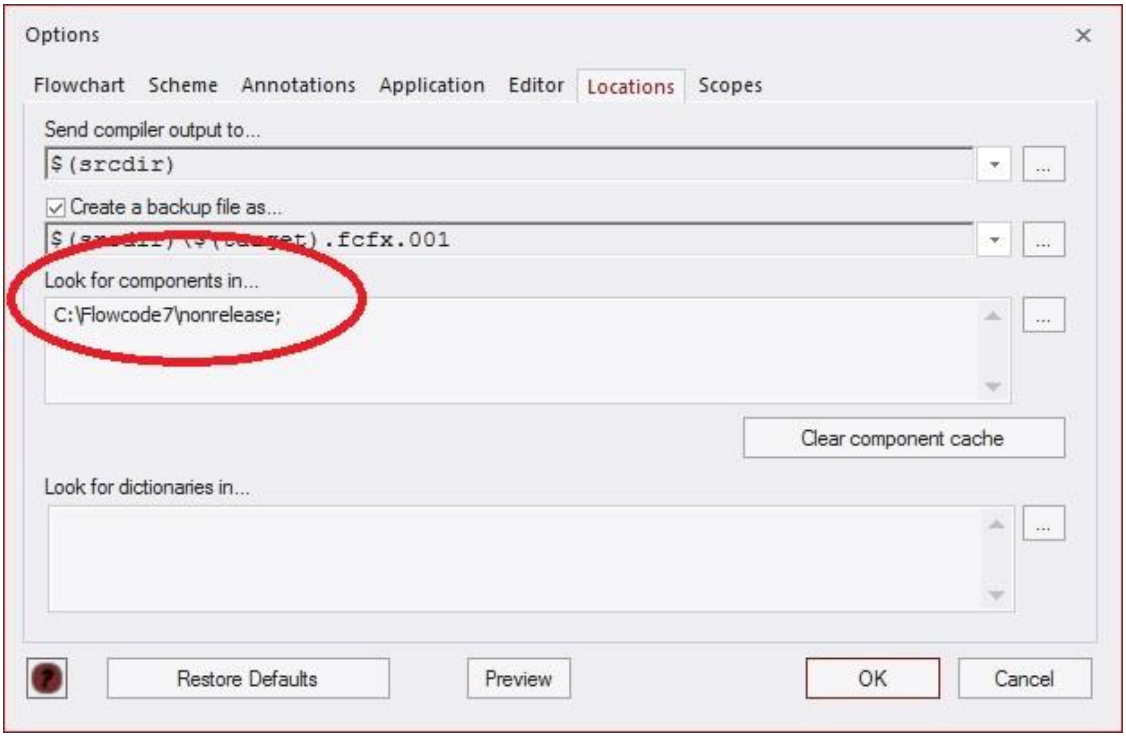
Then click the “OK” button to close the Component Management dialog.

If not already done, Save your project.

Now export this new component via the menu FILE->Export Component

It is important that you save any custom components to a directory other than the Flowcode components directory. Otherwise if you store them along with the Flowcode released components your custom ones might get deleted during an upgrade process.

Your custom components location directory can be added to the Flowcode search path by adding your directory in the dialog found at menu VIEW->Global Options



Close Flowcode and re-open to include your new component and create a test program.

Your new component should appear in the default “Misc” list, if not changed, otherwise search for it by name.

Advanced options

You can create an additional component Category for your components by adding it to the list found in the xml file "cmpntcats.xml" in your Flowcode installation components directory.

Using a simple text editor, add your category before the final </root> tag, for example:

<category name="User" hint="" icon="$(cmpntdir)\catimgs\User.png" tooltip="" />

Edit your component source project file to use this category by going to the component configuration dialog FILE->Component configuration. Change the component category in the “Setup” “Standard” tab.
Save the project and export the changes from the menu FILE->Export Component