Custom Component

Creation tutorial

Subject: DS3231 RTCC

General Information

Since flowcode v6 came out, the possibility to easily create components is true. However there are still some unclarities about the proces of creating components. Thats why I'm creating this tutorial because I think that creating components is easy. Of course you need to know how you handle things but that is what this tutorial is all about.

In this tutorial I want to explain and discus how to create a component from scratch. Or you just want to create a component just because you want to contribute to the flowcode community. Anyway, in this tutorial we are going to discus the following device: DS3231 this device is an extremely accurate RTCC (Real Time Clock Calendar). The DS3231 uses the I2C bus for communication. This component needs to provide all the features of the device so the end user can get the maximum of the device.

Gathering Information

We are going to create the component based of the data sheet from the device. There for its recommended to either have dual monitors, split screen or just to print the data sheet. I prefer to use dual monitors but that choice it up to you to make.
In the data sheet, the features of the device are listed (most of the time), for the DS3231 these features are:

- Real-time Clock Counts Seconds, Minutes, Hours, Day, Date, Month, Year + Leap Year
- Two Time-of-day Alarms
- Programmable Square-Wave Output
- Digital Temp Sensor Output
- Aging Trim Adjust register

So our component needs to be able to handle all these features, this lets us determine the macros that we are going to need for our component:

- ReadTime
- WriteTime
- ReadDate
- WriteDate
- SetAlarm1
- SetAlarm2
- GetTemperature
- CheckStatus —> because there is a status register available
- SetAging —> This optional be does not need to be implemented simply because a normal (hobbyist) user doesn't care.

So when we are finished with this tutorial we have a component with at least 8 ”downloadable” macros.

The next step in creating this component is to setup our working environment, also we need to setup the component properties.

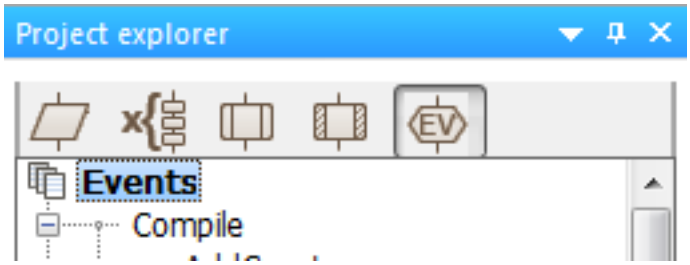
Configuring Component Properties

Before we can start creating the different macros, we need to create the component properties. The component properties lets the user change for example; the Square-Wave output frequency and the I2C bus settings.
So first we open a new flowcode program. In this new program you need to go to the panel properties and click the drop down arrow that appears when you hold the mouse on the left side of the properties bar.

Creating Property Events

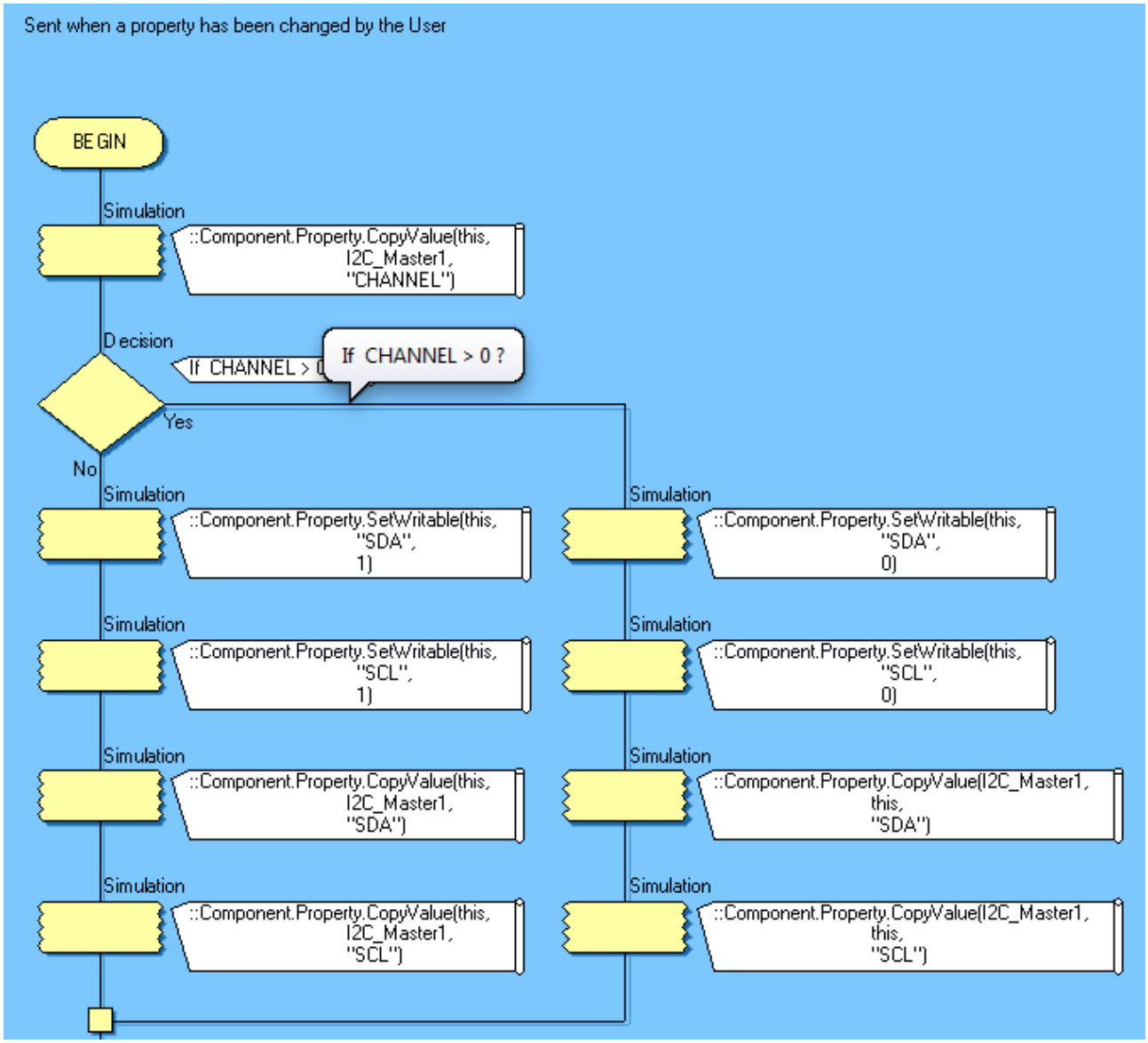
When you have done this and your are satisfied with the result so far we can begin to create some actual code. The next step is to make sure when the user of the component changes any of the bus related settings then this change needs to reflect to the base component in order to work correctly ones compiled.
To do this we need to create an event macro. This can be done by going to the project explorer of flowcode. At the top of this windows you can click the event button.

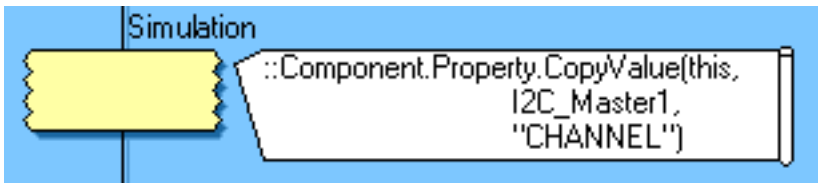
As you can see, you have a lot of events to choice from, but we are only interested in the component related events. And we want to create an event every time the user changes a property of this component.



Therefore you need to double-click on the Property tag located under the Components section of the Event tree.

Ones you have done that, flowcode will automatically create a new macro named Ev_Property. In this macro we are going to create some code that is going to be executed every time that the user changes o of the properties of the component.





All macros in the Ev_Property macro are Simulation macros. As you can see there are three parameters for the simulation function Property.CopyValue.

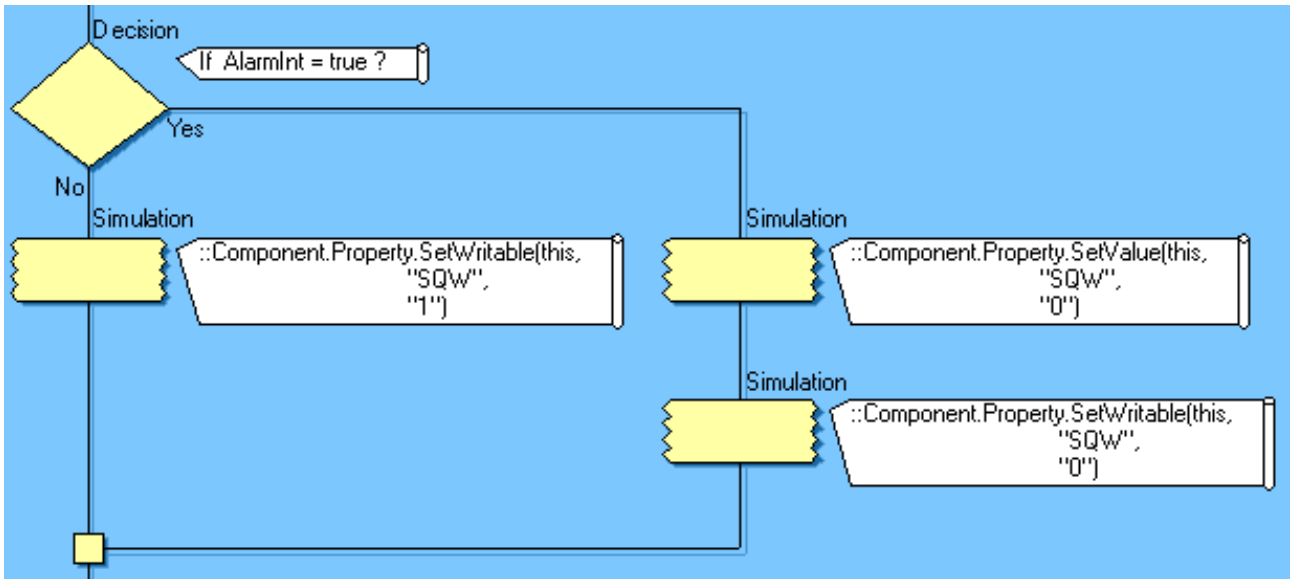
The first macro copy's a value from a source (this) to a destinations (I2C_Master1) and the value is CHANNEL. For instance if we choice to use the hardware channel for our I2C bus then this also needs to be applied (told) to the I2C_Master1 component.

So lets say we want to use the hardware channel for our I2C_Master1 component, normally the connections property of the I2C_Master1 component will be greyed out and set to not-writable. However our component does not know this. Therefore we need to make a decision if the Channel is set to zero (Hardware Channel), then we need to make the connection property not-writable en greyed out just like in the I2C_Master component.
Also we need to display the correct connections. So the first two simulation macros set the property SDA & SCL writable = False (0).

The last two simulation macros copy's the value from the property SDA & SCL into our component so that the connections of our component matches does of the I2C_Master1 component.

If channel is set to anything but zero then the opposite is true. The property SDA & SCL become writable and copy's the chosen port.pin to the I2C_Master1 component.
This methode can be applied to all other bus related settings.

The other thing we need to take care of is the conflict that is possible if the property Square-wave output Enable is set to anything but disabled and Alarm Interrupt Enable is set true.
This needs to be avoided because is can / will lead to faulty data written to the DS3231.
We need to create something when Alarm Interrupt Enable is set to true, then the property Square-wave Enable needs to be non-writable and default value needs to be set to, Disabled.



So when AlarmInt (variable linked to Alarm Interrupt Enable) is set to true (1) then we need to set the value of the Handle this (which is our component) with property SQW to false (0).
If we would only use the first simulation macro then the value of SQW is set to Disabled whenever we enable the Alarm Interrupt Enable to true. The only problem is that the user still can change the property SQW to a value he desires. To avoid this we need to set this property to Writable = False. Which can be seen in the seconds Macro.

Creating a Downloadable Macro

A downloadable macro is what the name suggests, a macro that will be downloaded or compiled into code for you device. For the complete ness of this tutorial I will explain how I created one macro. I will not describe all macros for this component because this doesn't contribute to this tutorial simply because all macro are build up the same way.

To create a downloadable macro just click on the macro icon in the project explorer window. You can already see some of the macros I have created the macro I will explain in depth is the ReadTemperature macro.

Before we can start writing code for this macro we need to have some parameters so the user can select what kind of return value he desires for his application.

Paramters:

- AddSign
  - True (1) = Will add a + / - sign to the output value
  - False (0)

- ValueType
  - Raw (0) = Raw sensor value will be returned to the user
  - Calculated (1) = The calculated sensor value will be returned to the user

- Unit
  - Celsius (0) = The calculated value will be returned in Celsius format
  - Fahrenheit (1) = The calculated value will be returned in Fahrenheit format

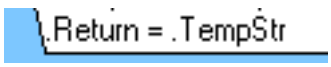
Return:

- String = Together with all the parameters the value will be returned as a string variable

The function of the read temperature macro will be to retrieve the temperature information stored in the DS3231. The information consists of a 10bit value, this value can be read from the registers 0x11 and 0x12. Since the DS3231 increments its register pointer automatically we just have to specify which register we want to start reading from is this case 0x11.

Since this is not a programming tutorials I will not go into the specifics of how to communicate with the DS3231.

With the different parameters that you have created you can create the output value accordingly. To assign the output value (which is string type) you just have assign you variable to the .Return variable.

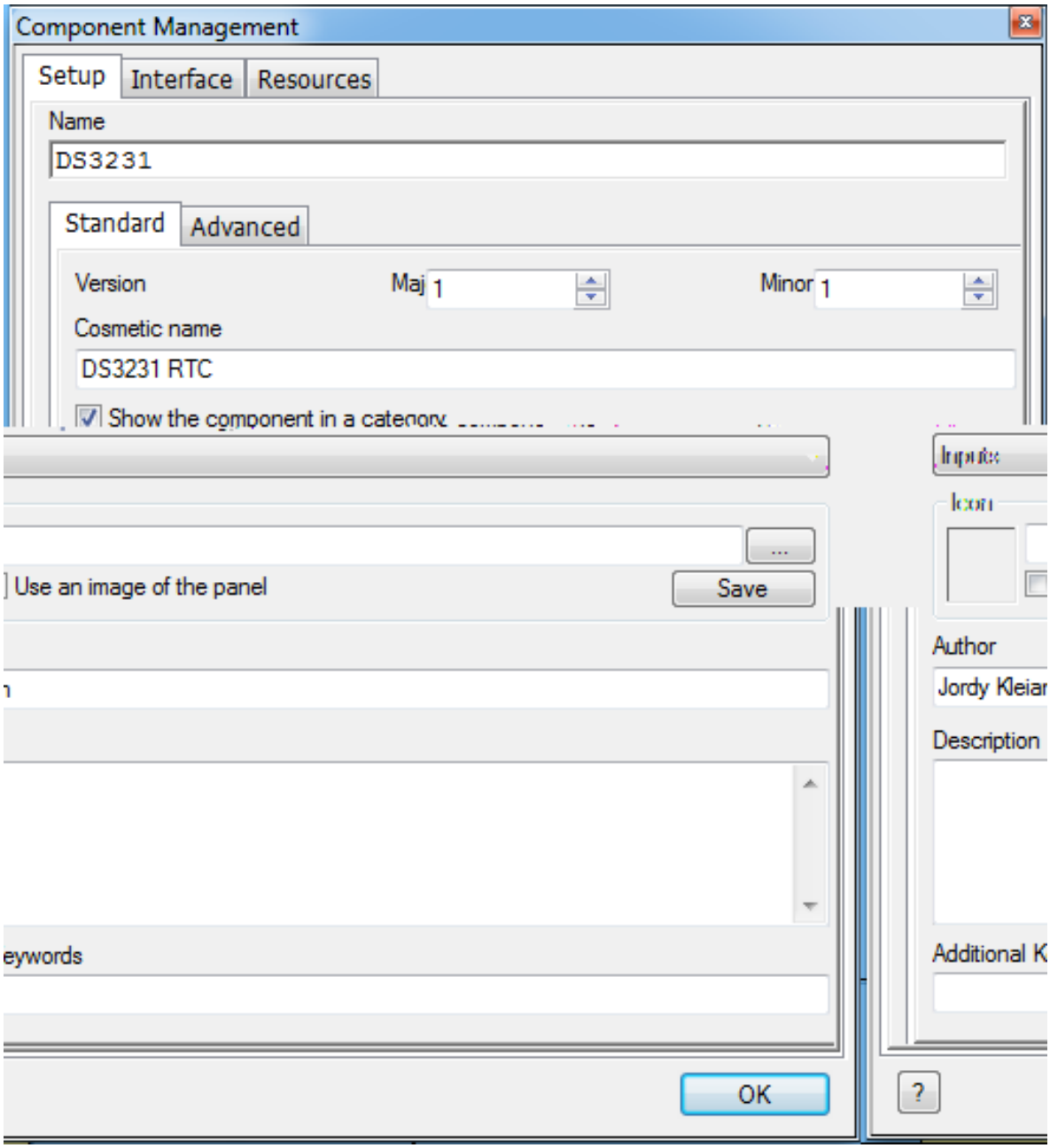


Export Your Component

When you have tested you component or want to look ad the behavior of the component when loaded as a component in flowcode, you need to export this component.
To export the component you have to take a number of steps.

1. Goto to the top of the screen and select File… and select Export Component… from this list

The following screen will popup,

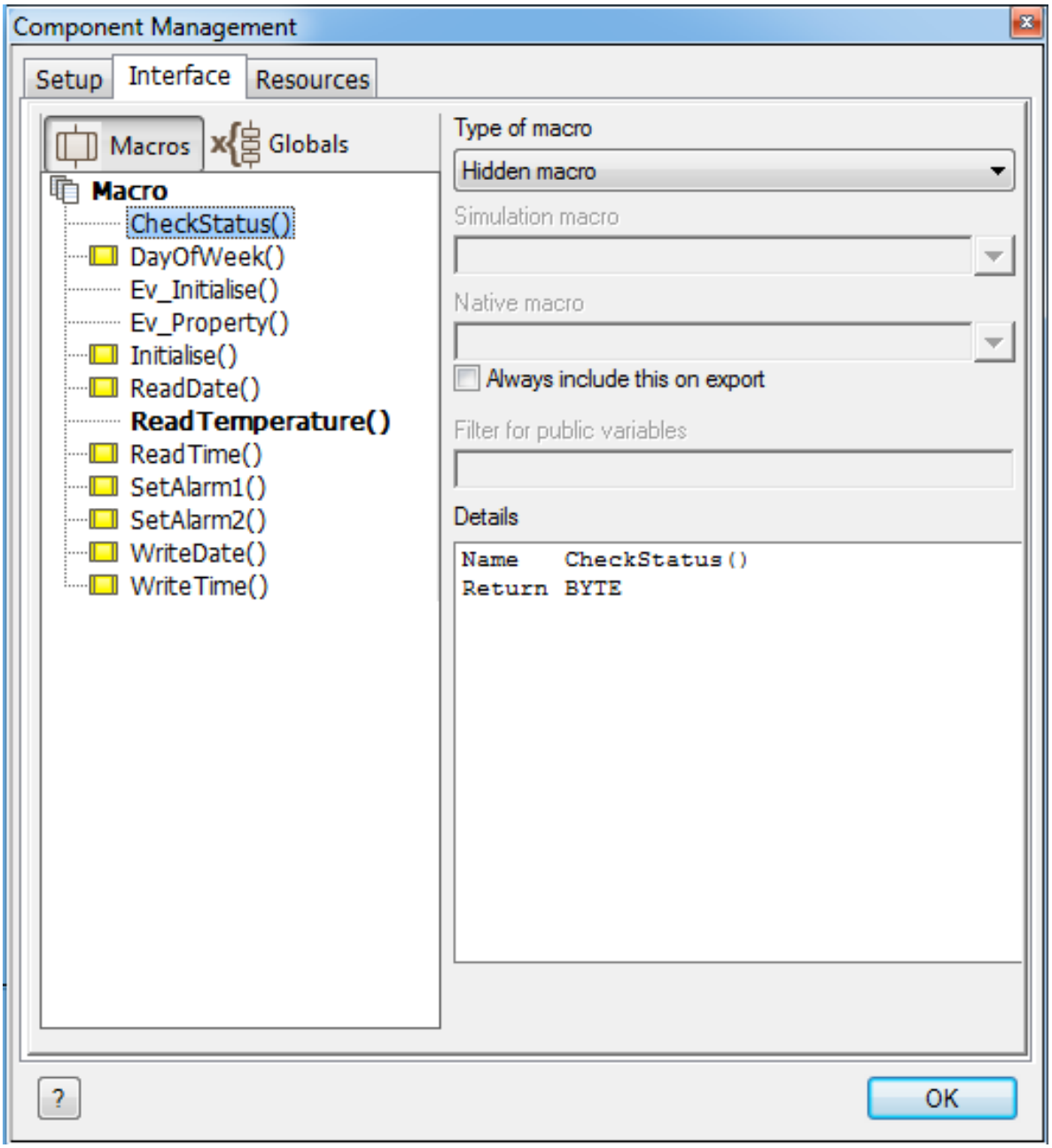


Most of the options speak for them self the 2 most importent settings are located under the tab Advanced (under the component Name) and Interface. what you see under the advanced tab at the bottom is GUID property. At the point when you want to create the first version of the component you will get a random GUID assigned, however when you want to export more version of the same component this GUID must not be changed. When this happens the component

already located in the component library of flowcode does not update to the latest version instead you will het to of the same component in the library but the version is different.

Then just before you want to export you component you want to specify which of the macros are made to be downloaded and which one needs to be hidden from the user.
For example the Ev_Property needs to be hidden from the user because it wont work on you microcontroller anyway, and it will probably report a compilation error.

For instance the macros such as ReadTemperature needs to be downloadable because you want this code to run on you hardware.



As you can see CheckStatus is a hidden macro because I have not completed this macro so the user of this component does not need to see this macro. All other macros with a yellow macro symbol are downloadable. These macro can be used in another flowcode program in the way you are used to.

Thanks For Reading

So folks thats all I have to say for now about creating component.
It's not hard to do, you just need to just do it, experiment with different settings and Event macros, to see what they are doing.

I hope that there are some part of creating component are discussed in this tutorial and that they where help full.
If there is anything in this tutorial that is not true, let me know and I will correct this ASAP.

It would be really cool to make a whole range of user-made components, I remember Benj talking about a component like web store kind of thing, which would be really cool.

Anyway, thank you for your time reading this tutorial, and I hope you are going to start soon develop you own components.

Regards,
Jordy Kleian
jordy101091

Additional Information

Component Creation Video - by Benj
Custom Component Directories - By JonnyW
Injector Components - by Benj