HCA Tech Note 107: Parameterized programs (updated 15-May-17)

HCA version 13 introduced a concept that can make creating a set of programs that handle a common task much simpler. This technical note shows a real world example of how the Parameterized Programs feature can be used. The example was taken from an HCA user's home but with some non-essential features reduced a bit to make it easier to follow.

The Problem

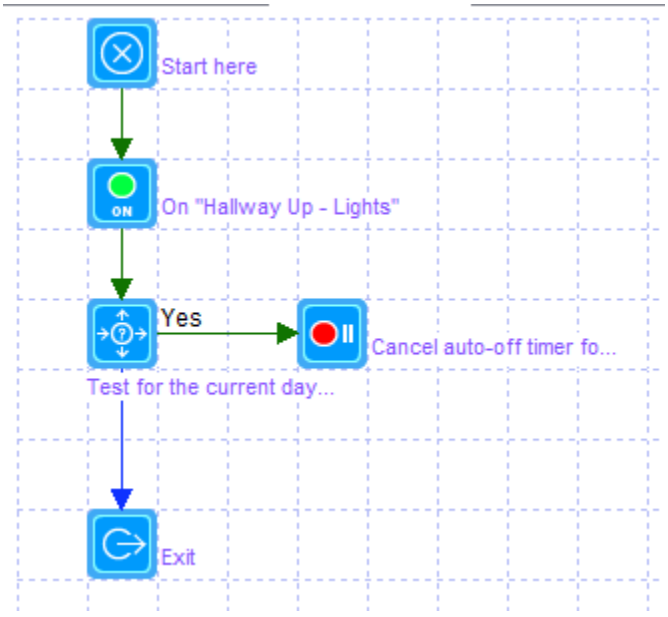
In this home there are two hallways – an upstairs and a downstairs. Both have controllable lighting and both have motion sensors.

The goal is to have the motion sensor turn on the hallway lighting and then on all days except Mondays and Tuesdays turn the lighting off after 10 minutes of no motion. On Monday's and Tuesday's no auto off should happen.

An OK Solution

The solution is very simple: The motion sensor acts as a trigger to a program and that program controls the light. An auto off specification is set on the light so that it goes off after 10 minutes. The program can be retriggered and that again controls the light and in doing so restarts the auto off timer.

One solution would be to simply create this program as part of the *Hallway Up* room and have it triggered on the motion sensor.



The Monday or Tuesday test is done using the HCA calendar and that's V13 feature for another technical note. More info on it is in the User Guide.

The motion sensor trigger starts the program, the light is controlled, and if Monday or Tuesday then the auto off is cancelled.

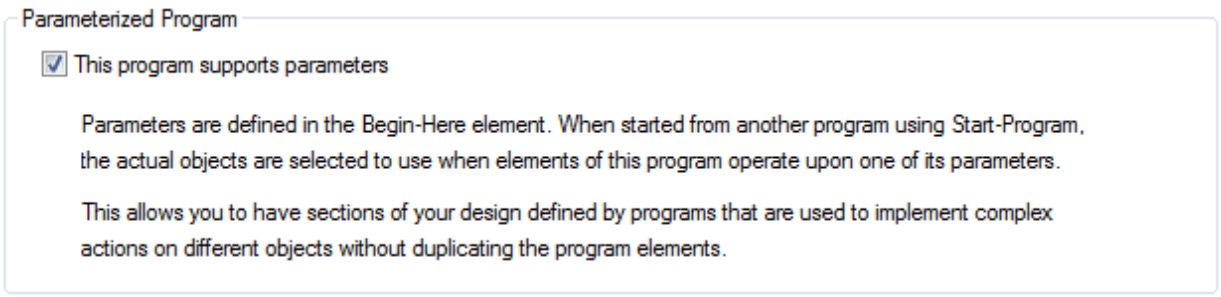
The next part of this solution would be to duplicate this program into the *Downstairs Hallway* room and change the ON and Auto-Off elements to use the light in that room.

But what if you wanted to change how the motion sensor and lights interact? Both programs must be updated. What if you wanted to have another room operate the same? The program would again have to be duplicated and updated.

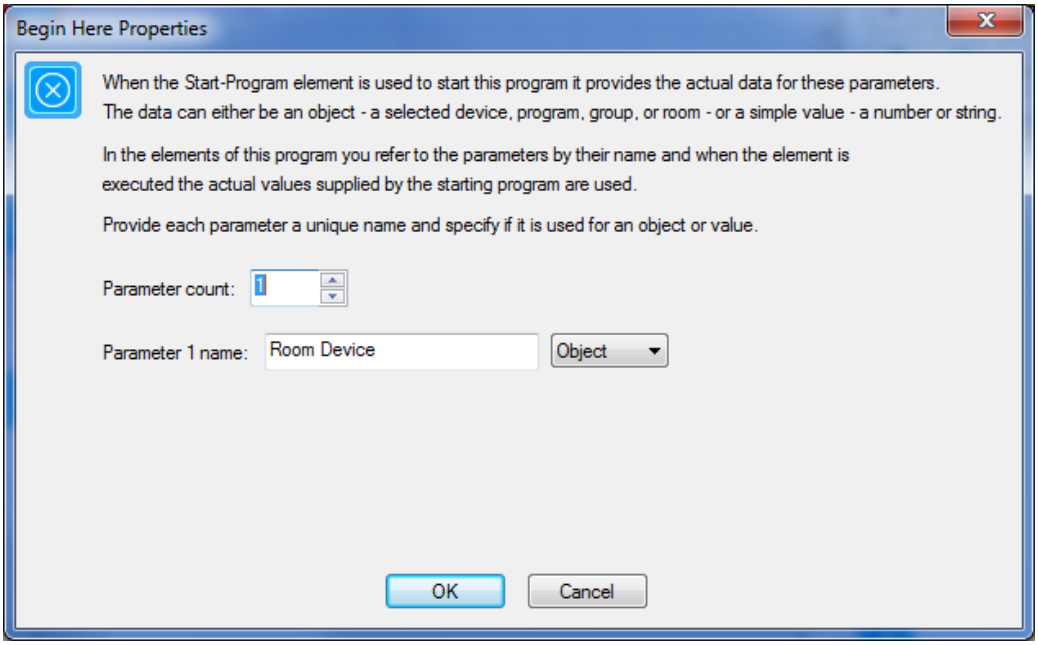
Improved Solution

A better solution is to implement one program to handle the auto off and tell that program what device to operate on.

The first step is to create a new program. This was done in the "Home" folder and is called "Handle Motion". Next, on the *Advanced* tab of the program properties, tick the option to enable parameters.

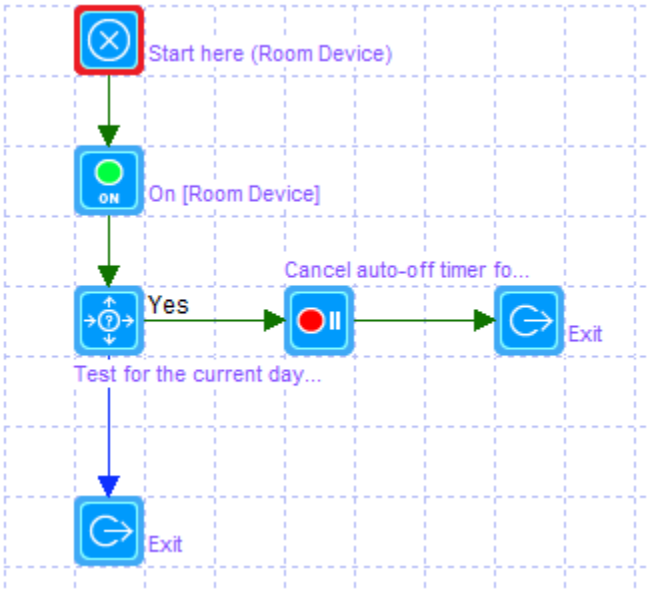


Once this option is enabled, on the Visual Programmer tab, the Begin-Here element now has properties that can be configured.

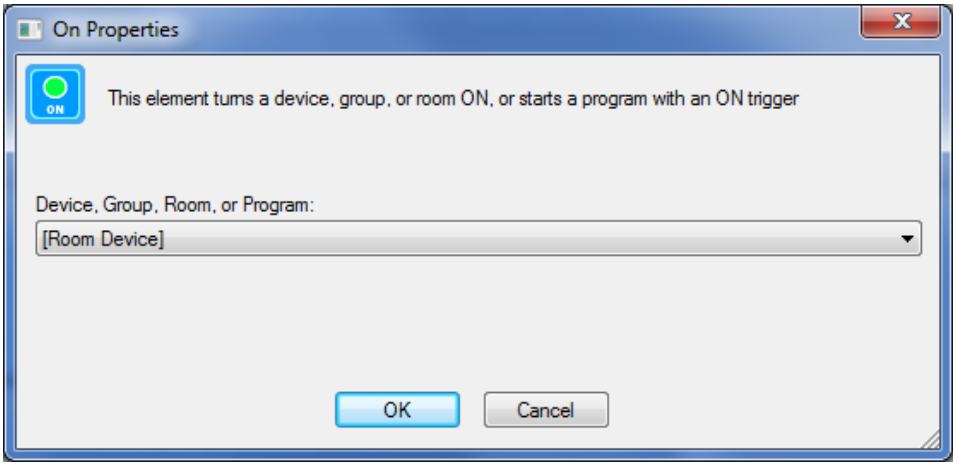


When the *Handle Motion* program is started, it is given the device to operate upon – in this example either *the Hallway Up – Light*s or the *Hallway Down – Lights* device. As a placeholder for whatever that device is, in this program we will refer to it as *Room Device.*

The rest of the program is the same as was created before with changes to the ON and Auto-Off elements.

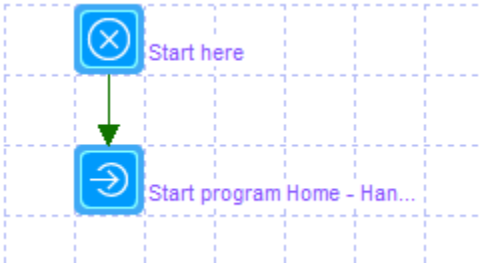


Instead of selecting the device to control in the ON and Auto-Off elements we instead select the name of the parameter - the placeholder - that will contain the device to control. When a program is marked as having parameters, the names for the parameters also show as selections in elements that operate upon devices or programs in addition to the devices and programs in the design as always. This is the changed ON element:



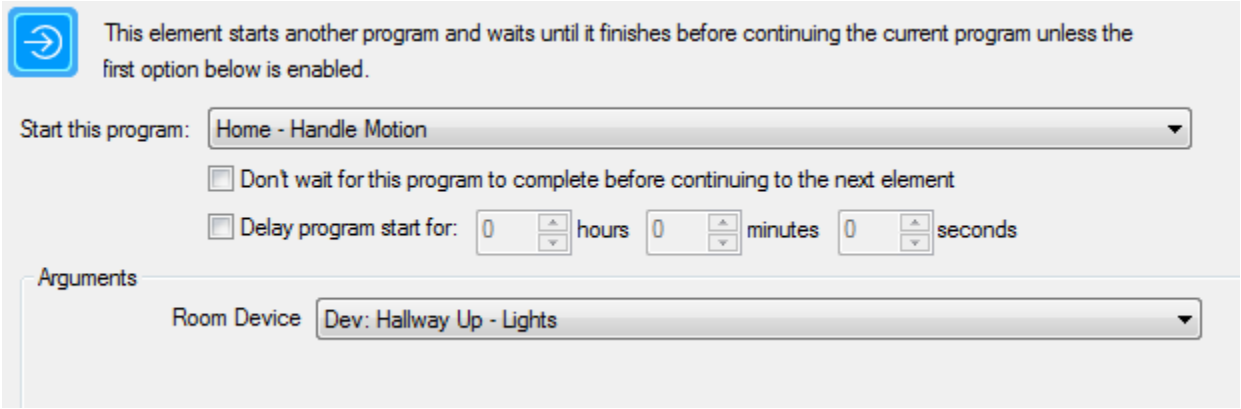
The same selection of the parameter name is in the auto-off element as well. And with these two changes, the *Home – Handle Motion* program is complete.

In the two Hallway rooms, we still need programs that start by a trigger from the room motion sensor:



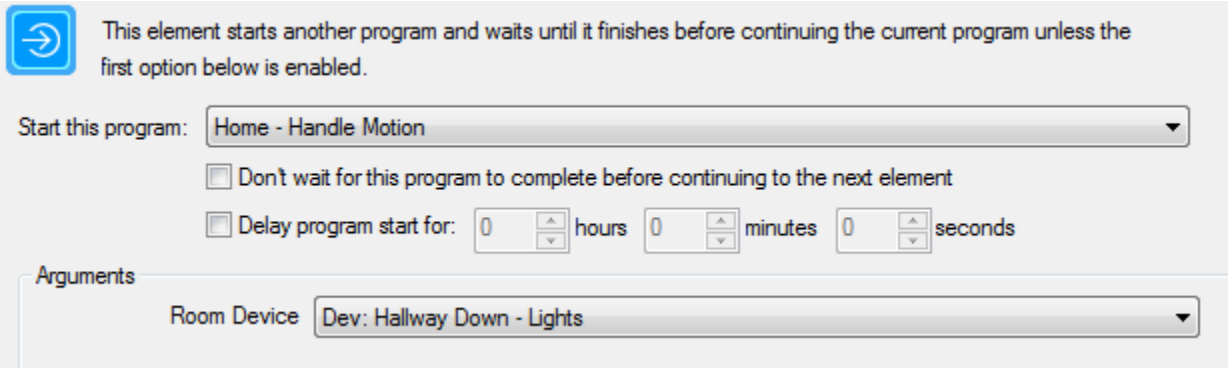
The program is very simple as you can see with only 2 elements. All it does is to start the program *Home – Handle Motion* and – this is the key – pass the device to be controlled.

The Start-Program element is configured as:



The Start-Program element starts the *Home – Handle Motion* program. Since it takes one parameter – the device to be controlled – the device to be passed in must be selected. For the *Hallway Up* room, the *Hallway Up – Lights* is the device to be controlled so it is selected.

All the remains now are to create another 2 element program in the *Hallway Down* room that is the same except this time the *Hallway Down – Lights* device is selected in the Start-Program element.

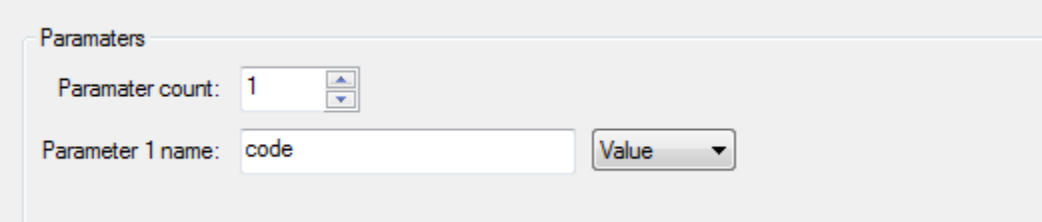


So what's the advantage of this?

The advantage comes in the fact that you now have a simple program *Home – Handle Motion* to update if your needs change. Also, if you wanted similar behavior for more rooms, then they all can use the same technique of using Start-Program on a parameterized program and passing it the device to work on.

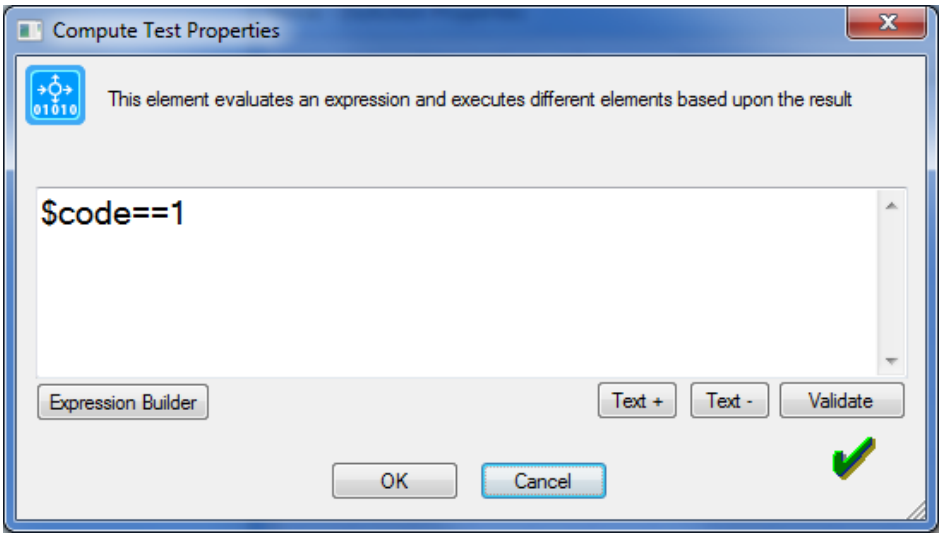
Values vs Objects

The above example was all about using an object – a device – as an argument to a program. But you can also use values – numbers, strings, date-time, Yes-No – as arguments.



For example, you may have a program that performs a set of related tasks and use it in several different situations. You can have the program except a value parameter that is a code – a number between 1 and n – that tells the program what task to perform. If you start the program with code 1 it does x, with code 2 it does y, etc.

Within the program you can use the Compute-Test element to check to see what that code is. For example:



In this way, you can test the value of the code passed as an argument and perform different actions as needed.

Want to know more about Parameterized programs?

The User Guide Programs chapter discusses these sorts of programs in much greater detail and show all the facilities and features of parameterized programs.

##end##