Control

Contents

Introduction 4
Overview 4
Collapse Arrow 6
Options 7
Run Type 8
View Routines, Vars, Errors 9
Open 9
Status 10
Menu 11
New Program 12
Actions 16
Sync Bundles 16
Upload 16
Download 16
Delete 17
Search 17
Search Options 17
Tools 18
View – Program Summary, Run Conditions 18
Program Summary 19
Run Conditions 20
Blocks 20
Prog/Task 21
Vars 23
Variable Settings 23
Write Level 24
Show Selected Variable Trace 25
Delete Selected Variable(s) 26
New Variable. 27
VAR FROM TEMPLATE 27
MULTIPLE VARS 30
POINT/TAG VAR 31
PROG/SHARED VAR 32

Select Task For Live Values 32
Introduction to the Control Engine 40
Overview 40
Library 40
abs 41
add 41
alarm 42
all 43
and 44
any 45
avg 46
boolLatch 47
callRoutine 48
cmpEq 49
cmpGt 50
cmpGtEq 50
cmpLt 50
cmpLtEq 51
cmpNotEq 51
comment 51
count 52
date 53
dateTimeAdd 53
dateTimeSub 55
dateToNumber 55
dateToString 56
delayedOutput 56
div 57
dtToDtAdd 58
dtToDtSub 58
email 59
Email Alarm 60
gridSelectCol 62
gridSize 62
if 63
Toggle Else Block 65
inRange 66

isEven 67
isOdd 67
isType 67
jobRun 68
leadLag 69
leadLagStageSelect 70
limit 71
max 72
min 72
modulo 73
mult 74
neg 75
not 75
now 76
or 76
PID 77
random 78
reset 79
return 80
set 80
    Toggle Timer 82
setExternal 83
sineWave 84
siteDateTime 85
staging 86
strCapitalize 87
strConcat 87
strContains 88
strDecapitalize 88
strEndsWith 89
strFormat 90
    Format() 91
    Bool.toLocale 91
    Int.toLocale 91
    Float.toLocale 92
    Date, Time & DateTime toLocale 92
strLower 93

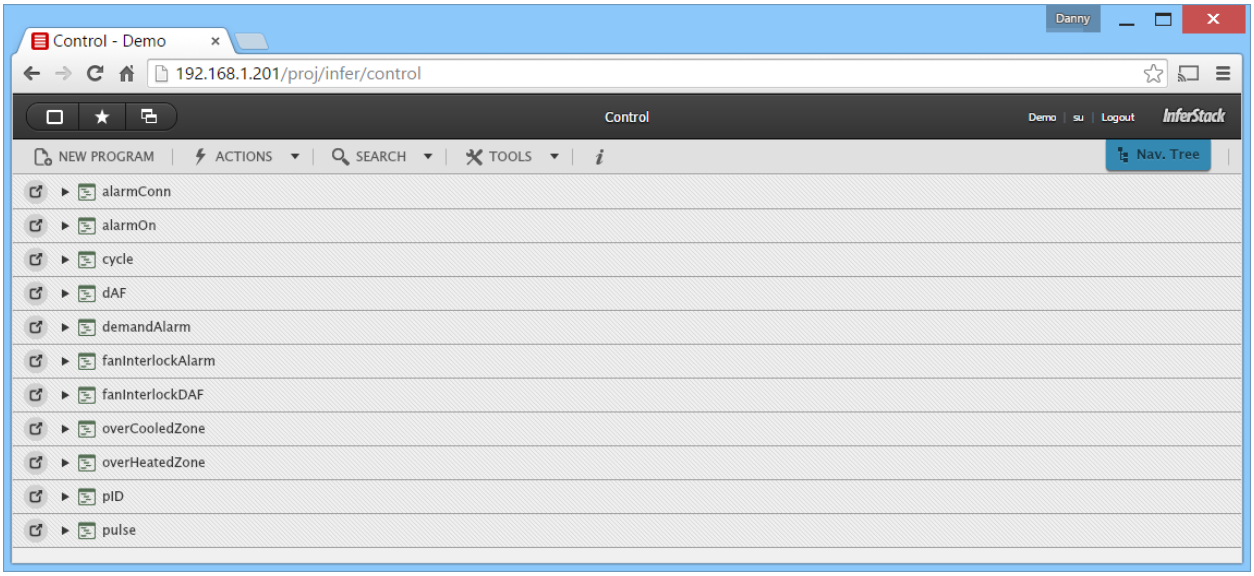
strMacro 93
strReplace 94
strSize 95
strStartsWith 96
strUpper 97
sub 97
threshold 98
time 99
timeToMinutes 99
timeToTimeAdd 100
timeToTimeSub 100
wait 100
weatherConditions 101
weatherInfo 101
weatherSunrise 102
weatherSunset 103
xor 103
Examples 104
Alarm On 104
Demand Monitoring and Email Value 105
Demand Response 106
Enumeration 107
Monitor Connectors 108
Oscillate 109
Over Cooled Zone 109
Staging 110
ahuStage 110
vavStage 111
Troubleshooting 111
Reference 113

Introduction

This is an introduction to the Control app of InferStack. We will cover how to access it through the graphical interface, and use it to develop and debug bLineLogic programs. Reference information is also provided for further study.

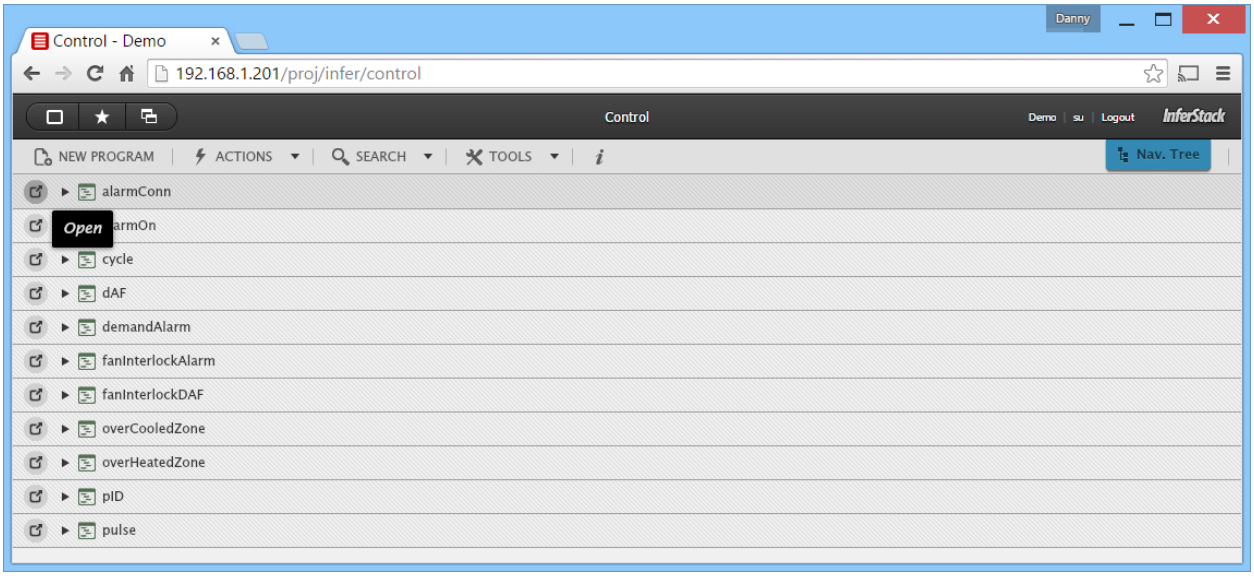
Overview

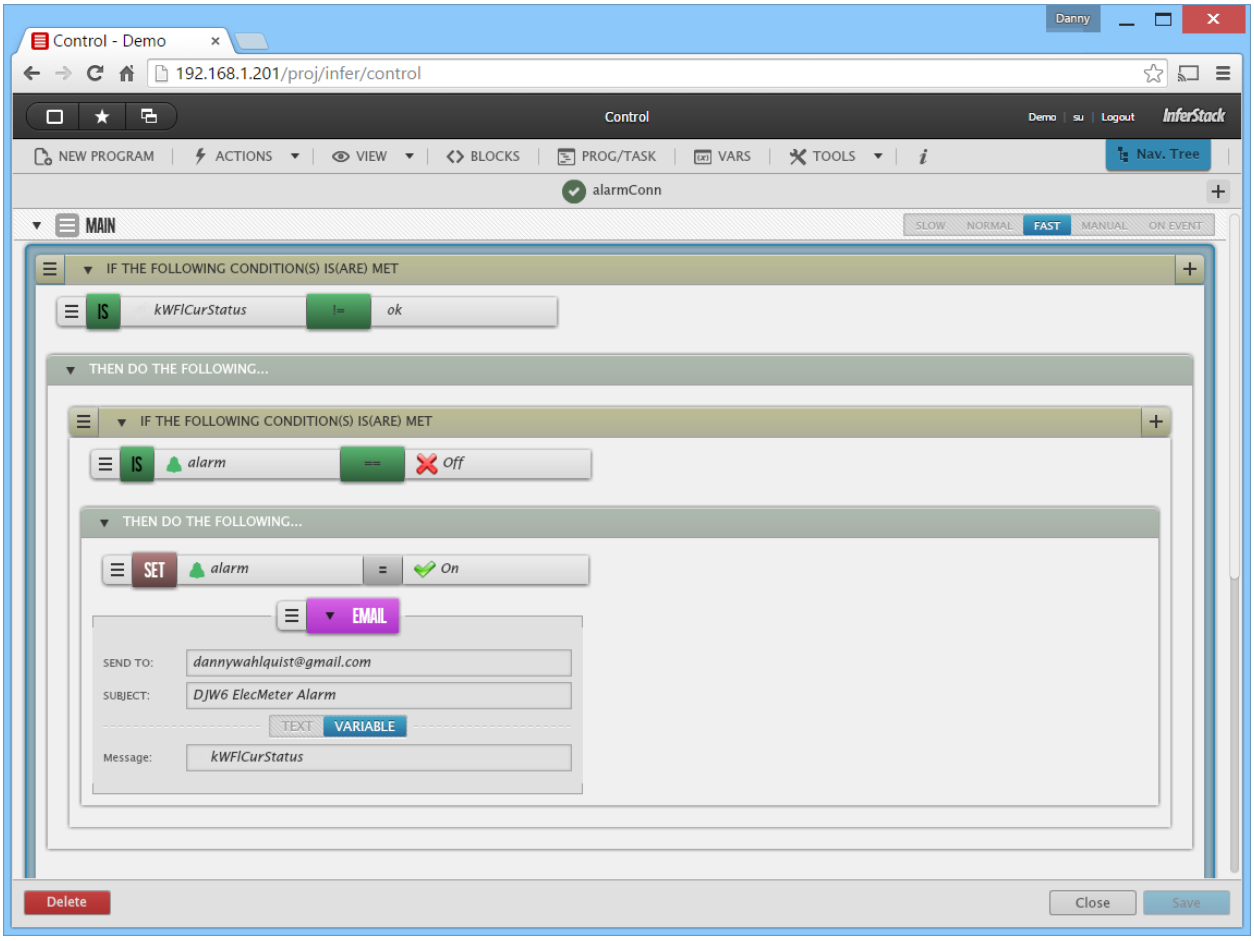
When you select the Control app, you are presented with a list of programs if any have been defined.



The menu at the top allows you to create a New Program, perform Actions, Search and Tools.

The Open icon to the left allows you to view a program.

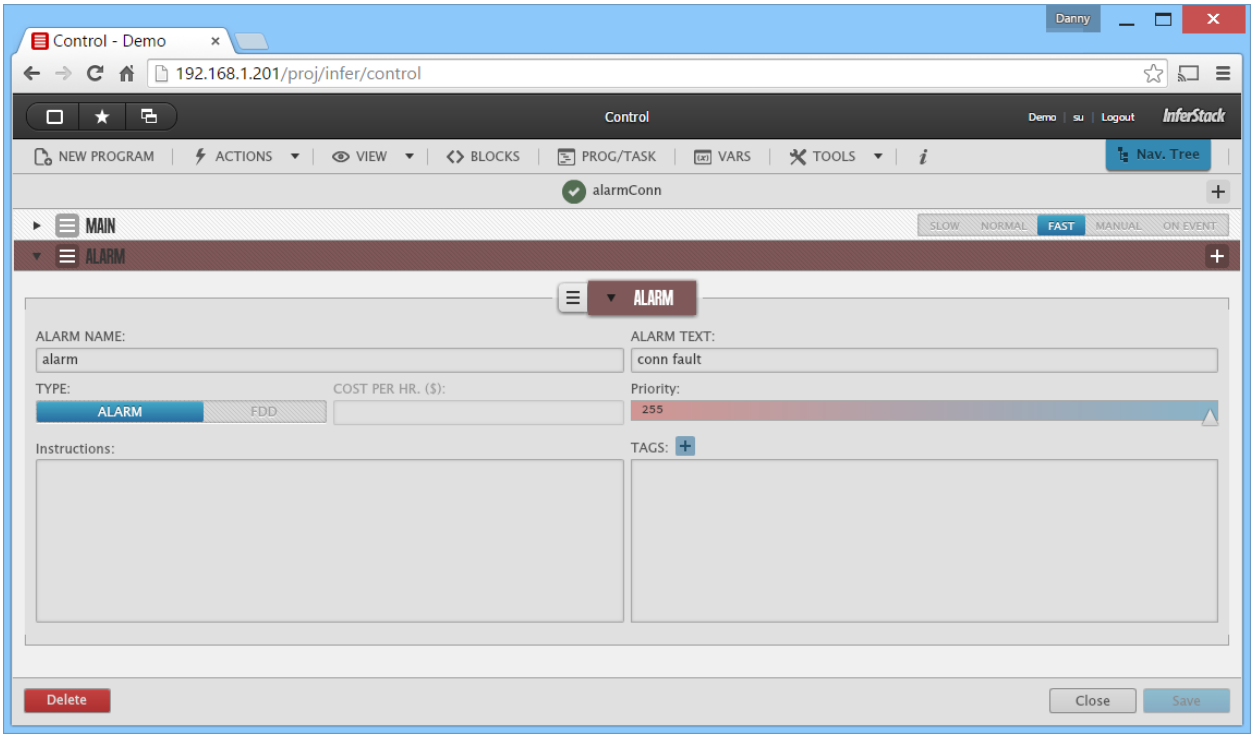




Once you have a program open, the top line will include a collapse arrow, an options icon and how the program will run (Slow, Normal, Fast, Manual or On Event).

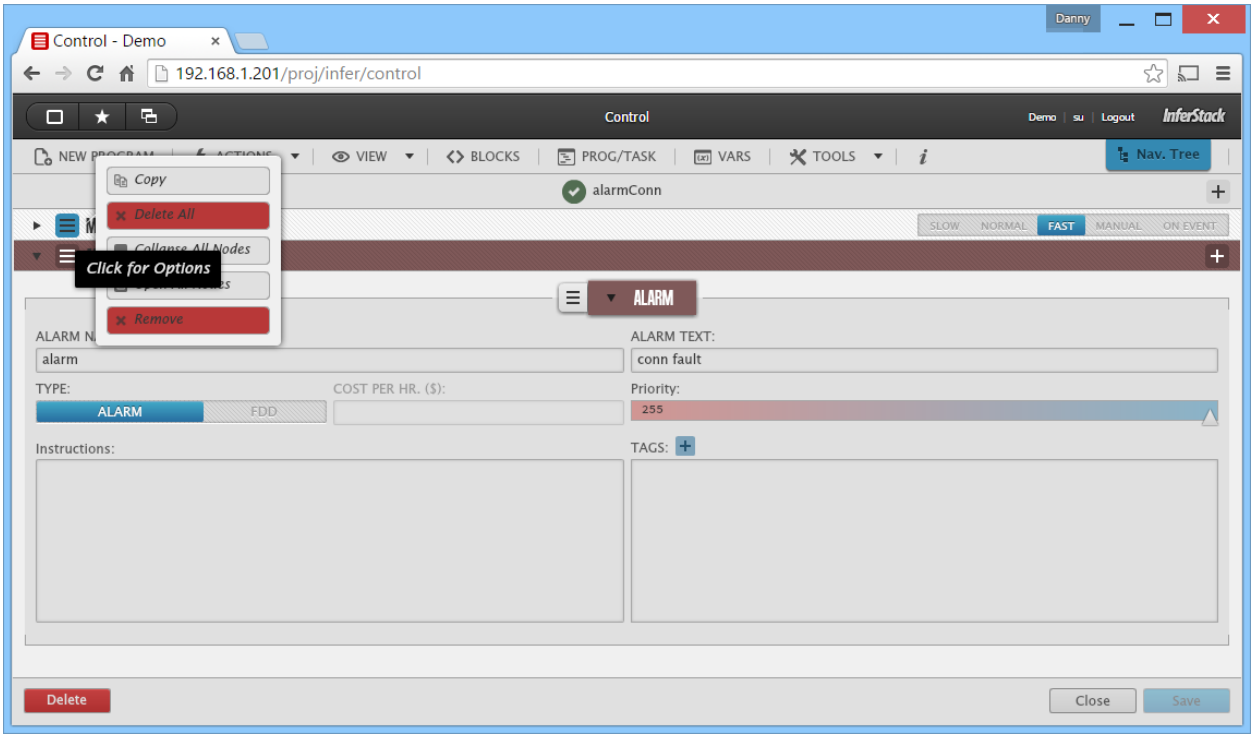
Collapse Arrow

The collapse arrow allows you to view or hide the lines of the program.



Options

The Option icon lets you access Options.



Press it to choose options that may include Copy, Delete All, Collapse All Nodes, Open All Nodes and Remove.

Run Type

The run type defines how the program will run (Slow, Normal, Fast, Manual or On Event).

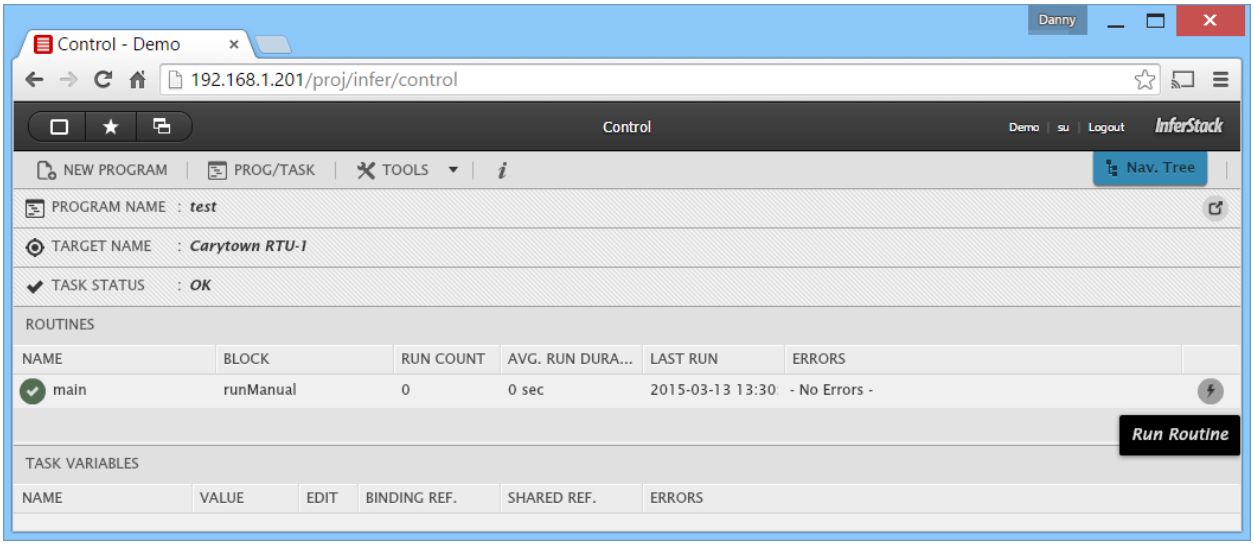
Slow, Normal or Fast

These speeds may be configured as tags on the ControlExt rec as Numbers tags:

- 'fastPeriod' or defaults to 100ms
- 'normalPeriod' or defaults to 1sec
- 'slowPeriod' or defaults to 10sec

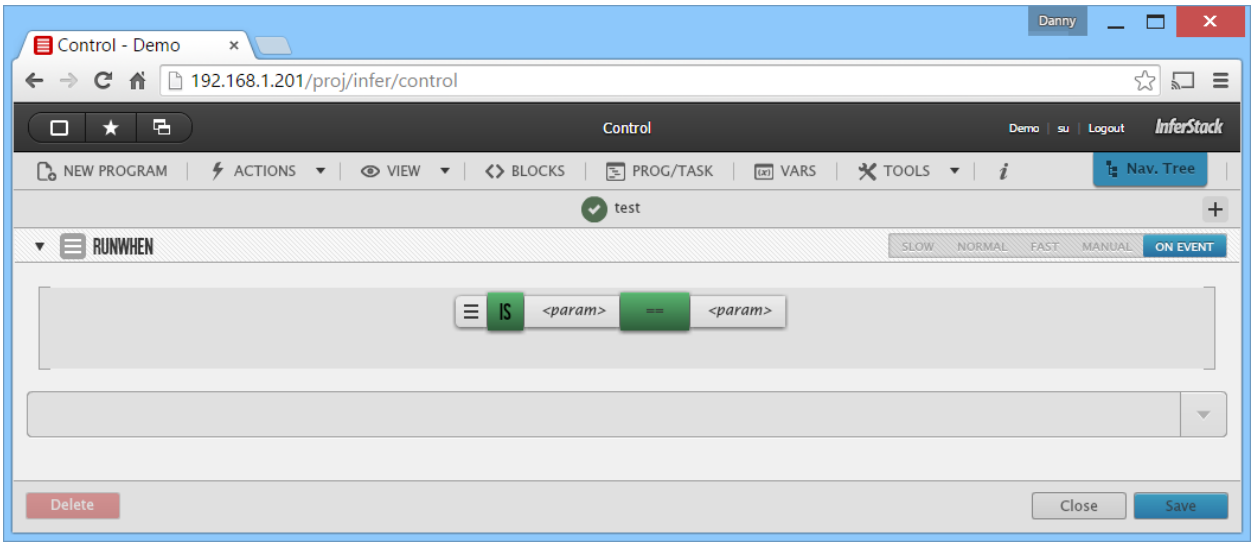
Manual

Manual lets you run the program when you command it to run.



On Event

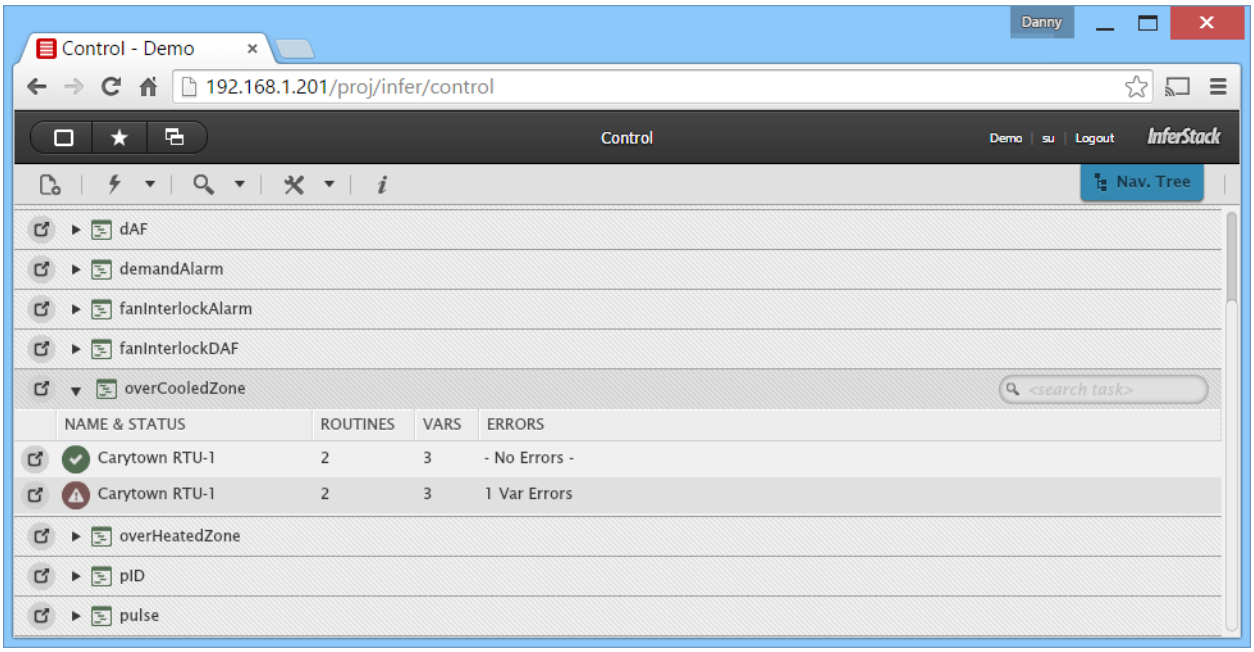
On Event lets you use a parameter to determine when the program will run.



Press Close in the lower right to close the program to return to the program listing.

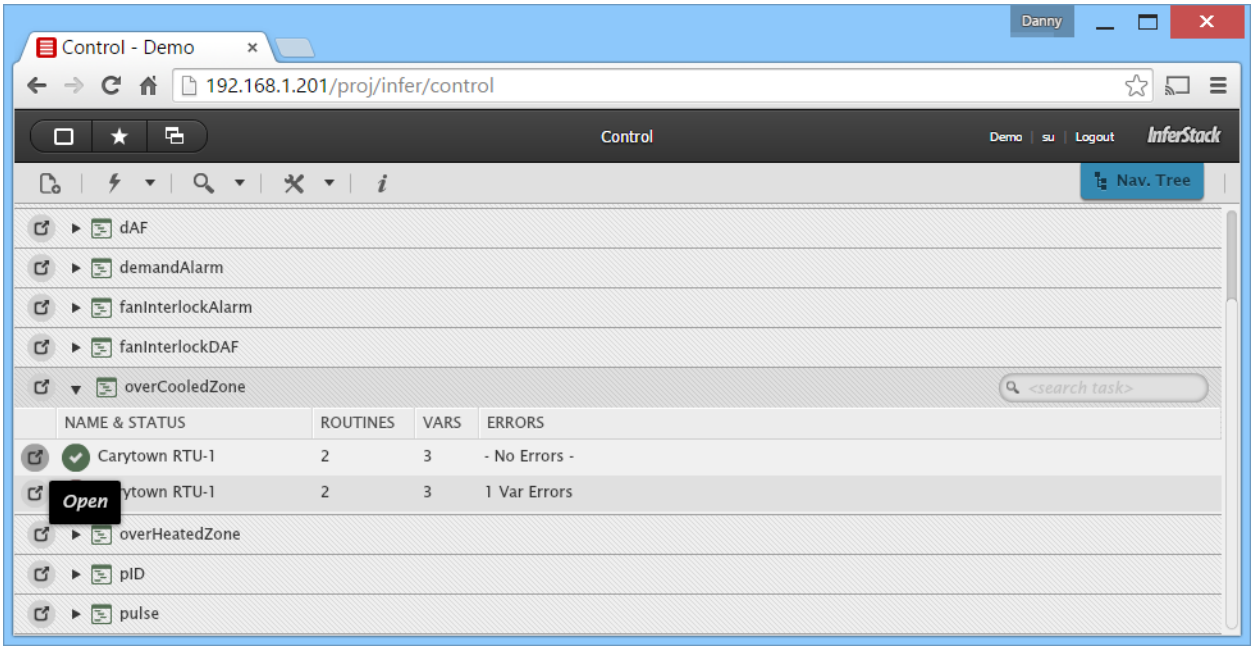
View Routines, Vars, Errors

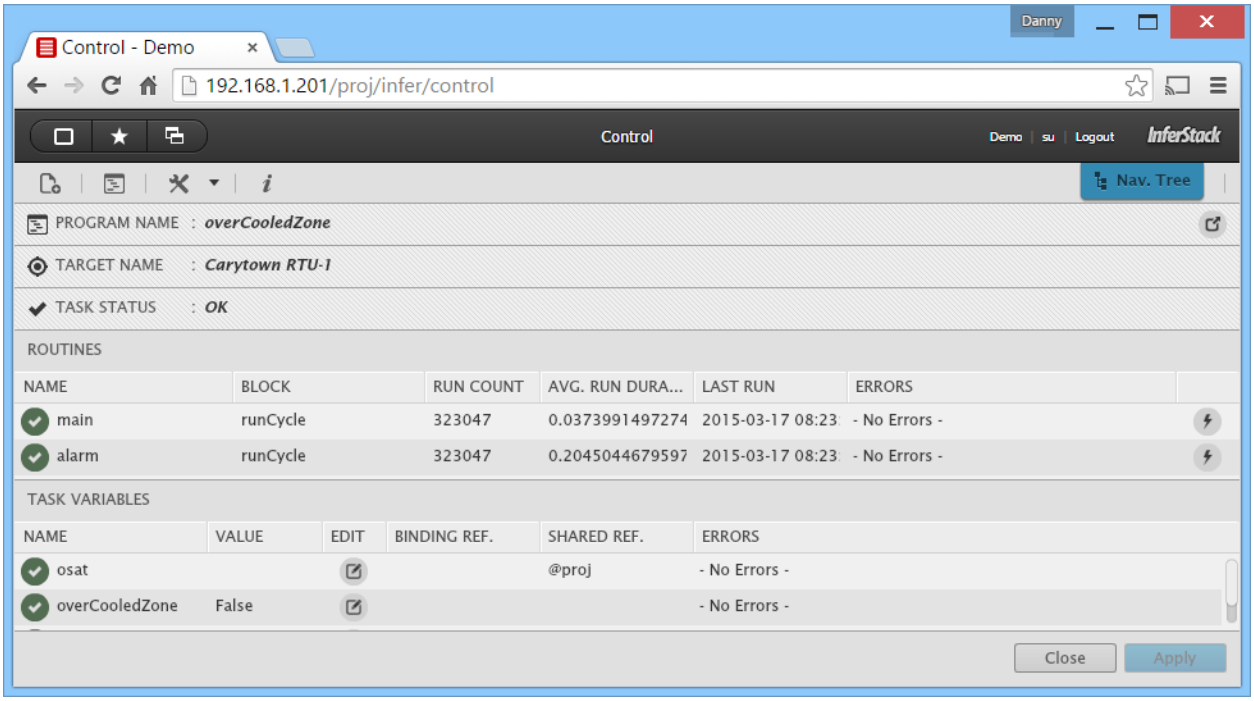
The small right arrow icon next to the Open icon allows you to see the routines, variables and errors.



Open

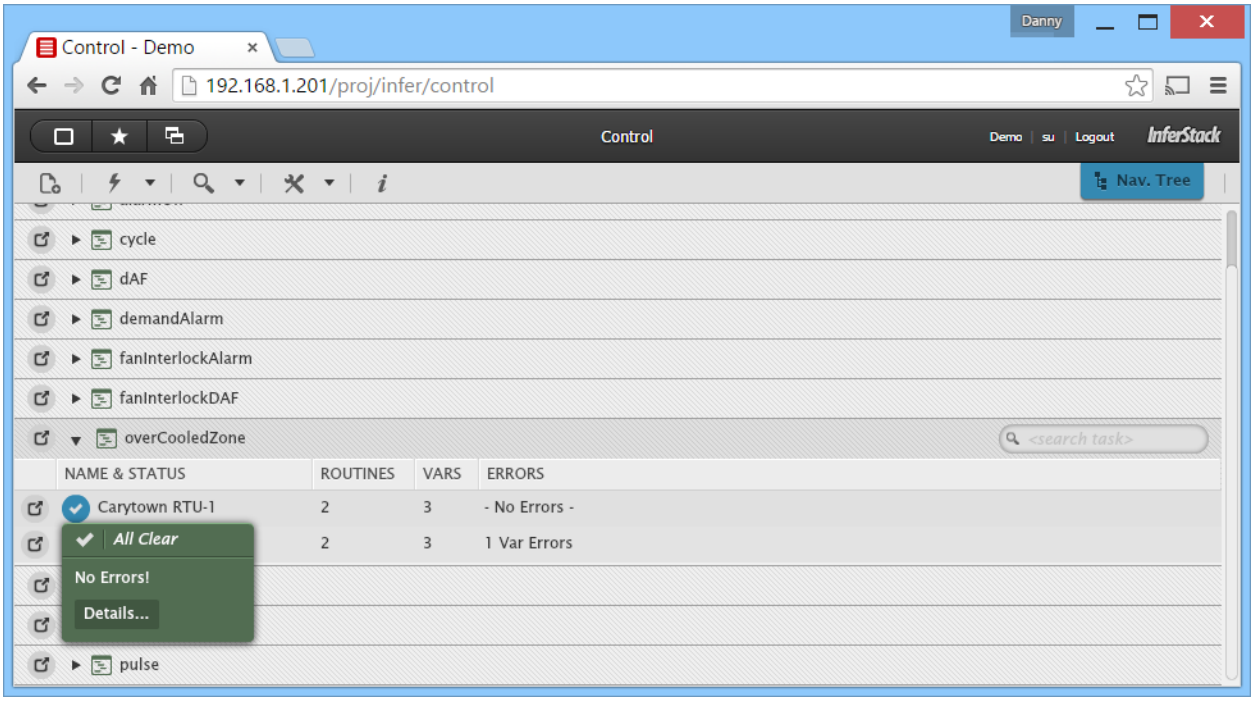
You can open the routine.





Status

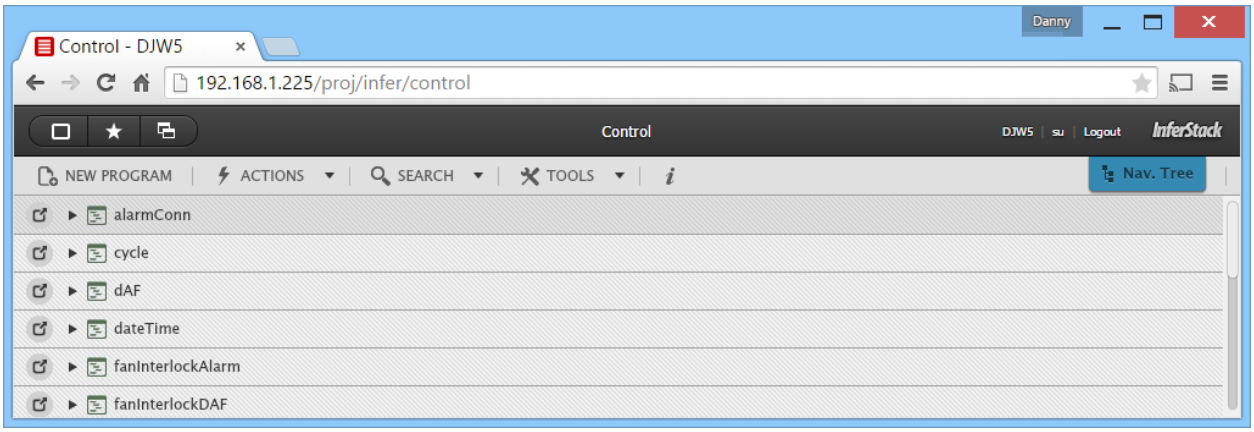
You can view status including details.



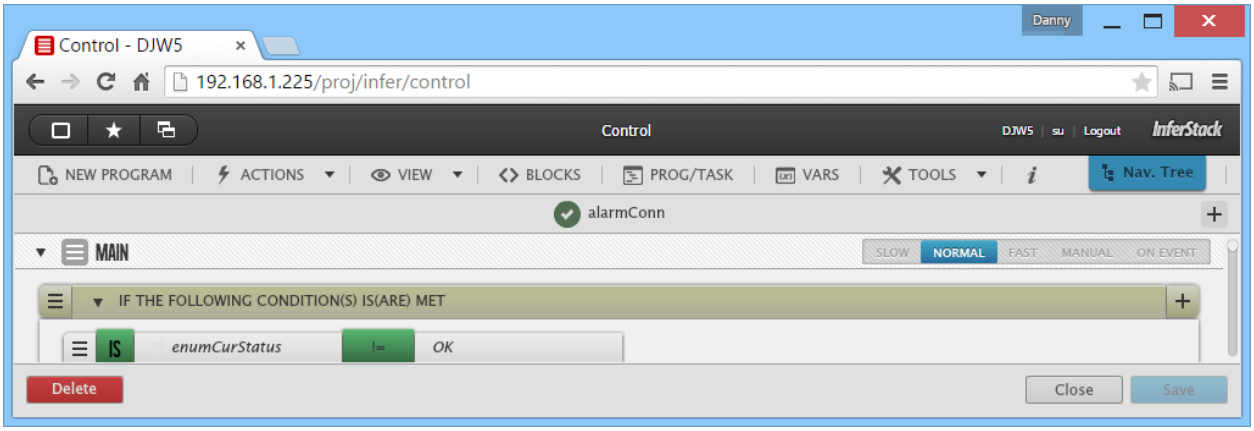
Press Details to view them.

Menu

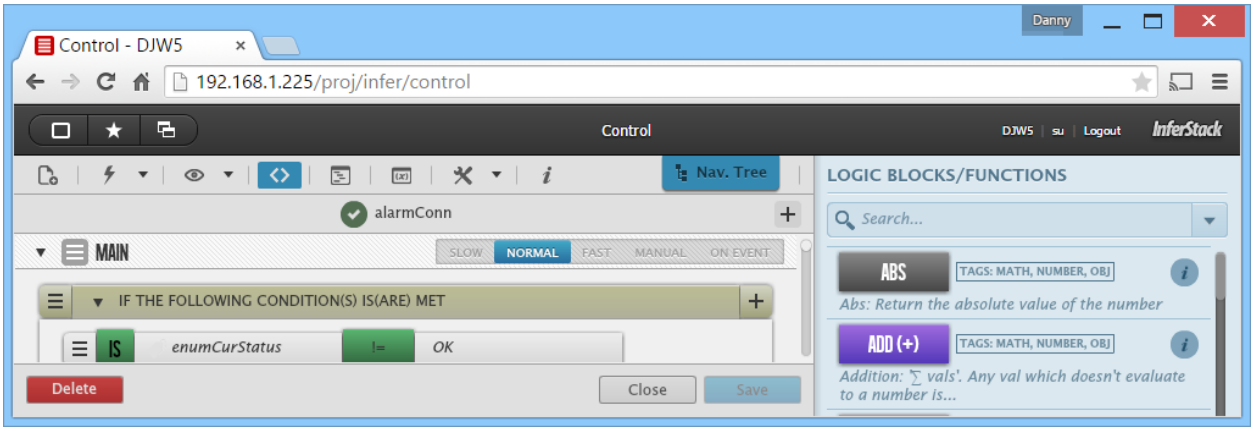
The Menu changes based on a number of factors.



If you open a program, you get other menu items.

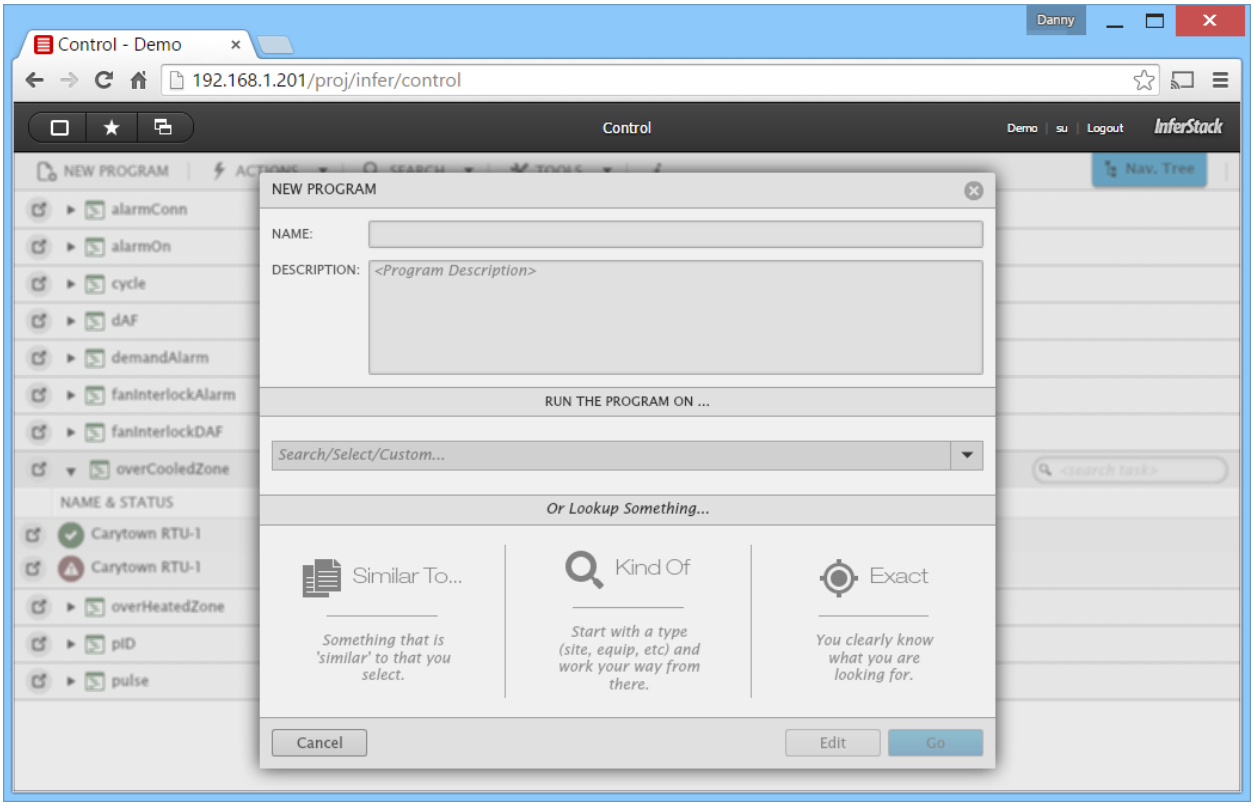


Also, if your window is too small to show the complete menu (such as with blocks open below), only the icons are shown.

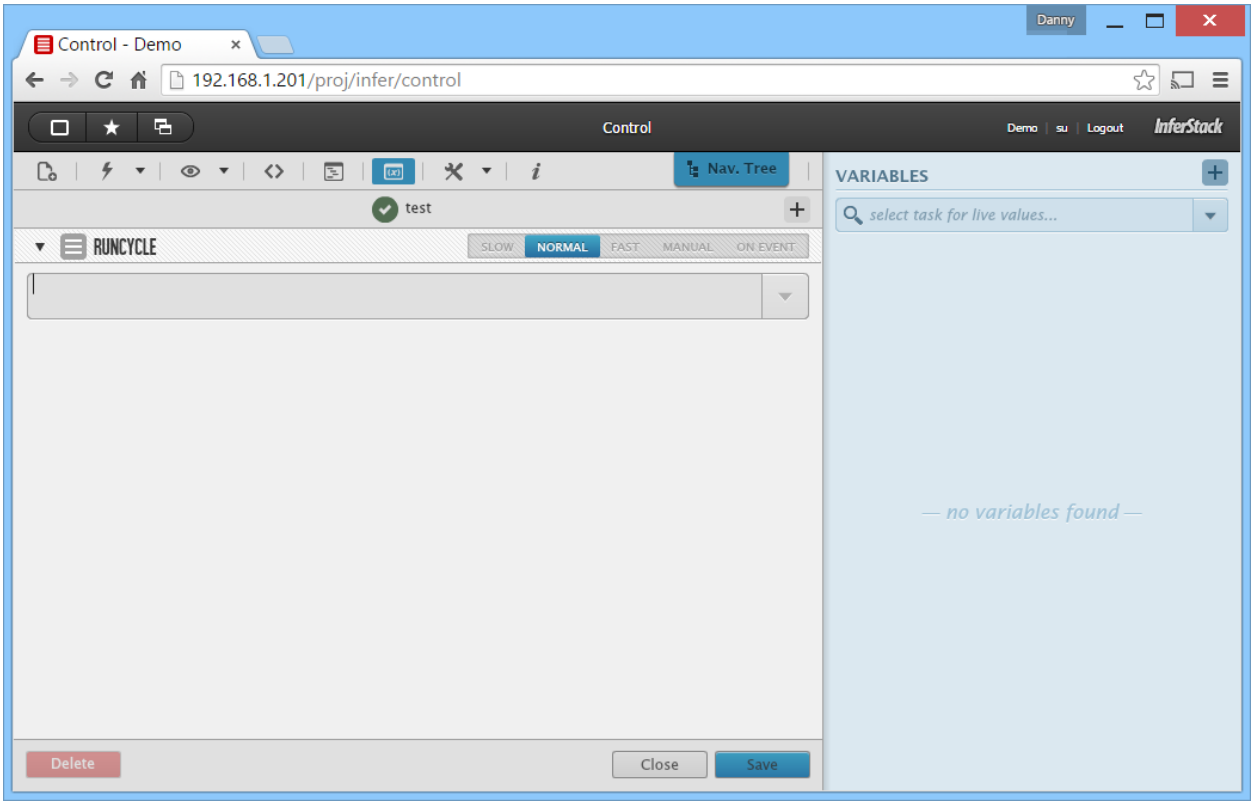


New Program

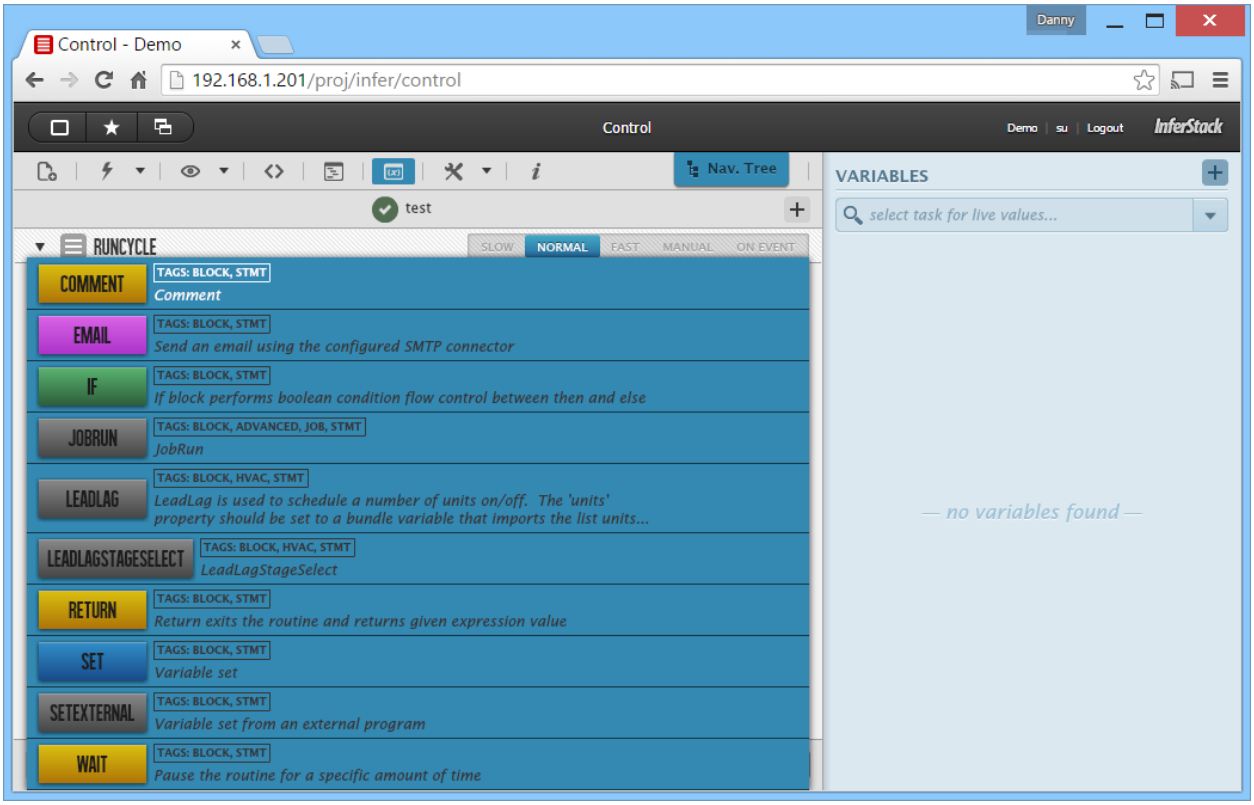
New Program lets you create a new program.



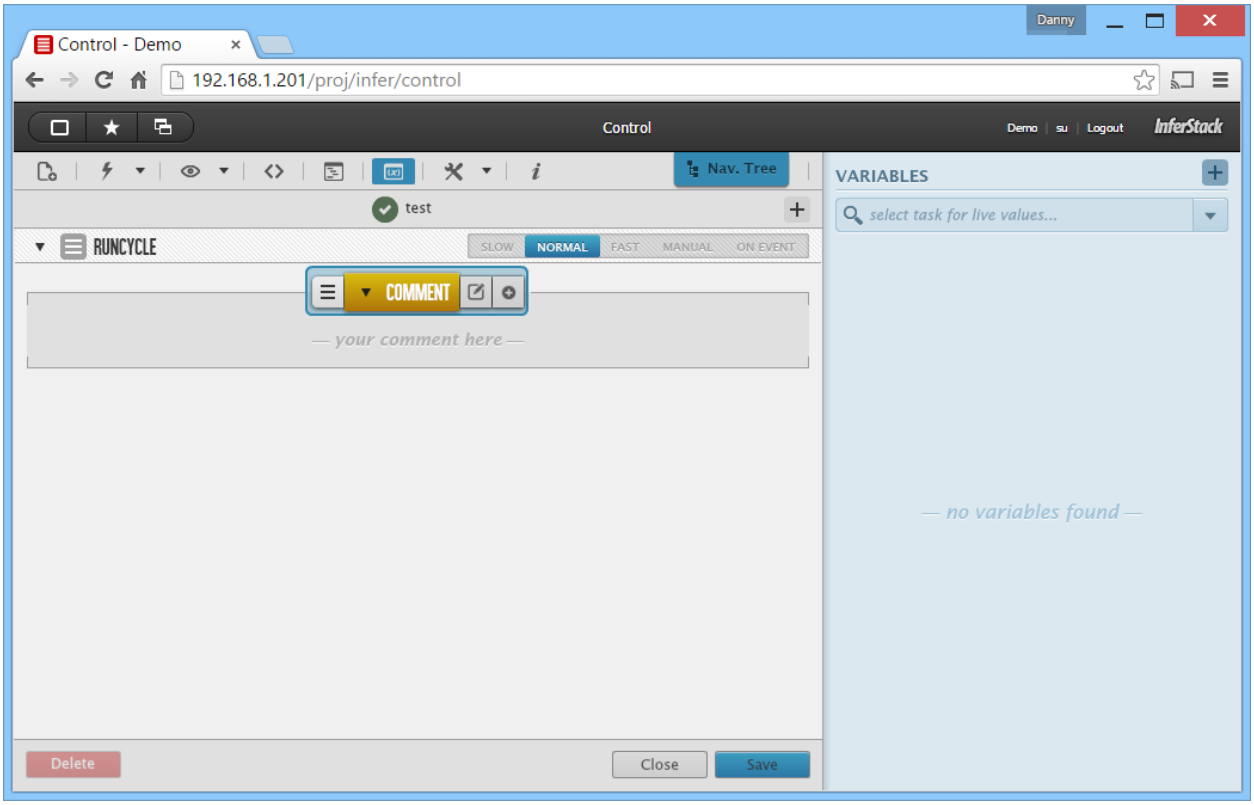
You must provide a Name and can provide Description, AND tags, OR tags and NOT tags.
Press Create to create a new program.



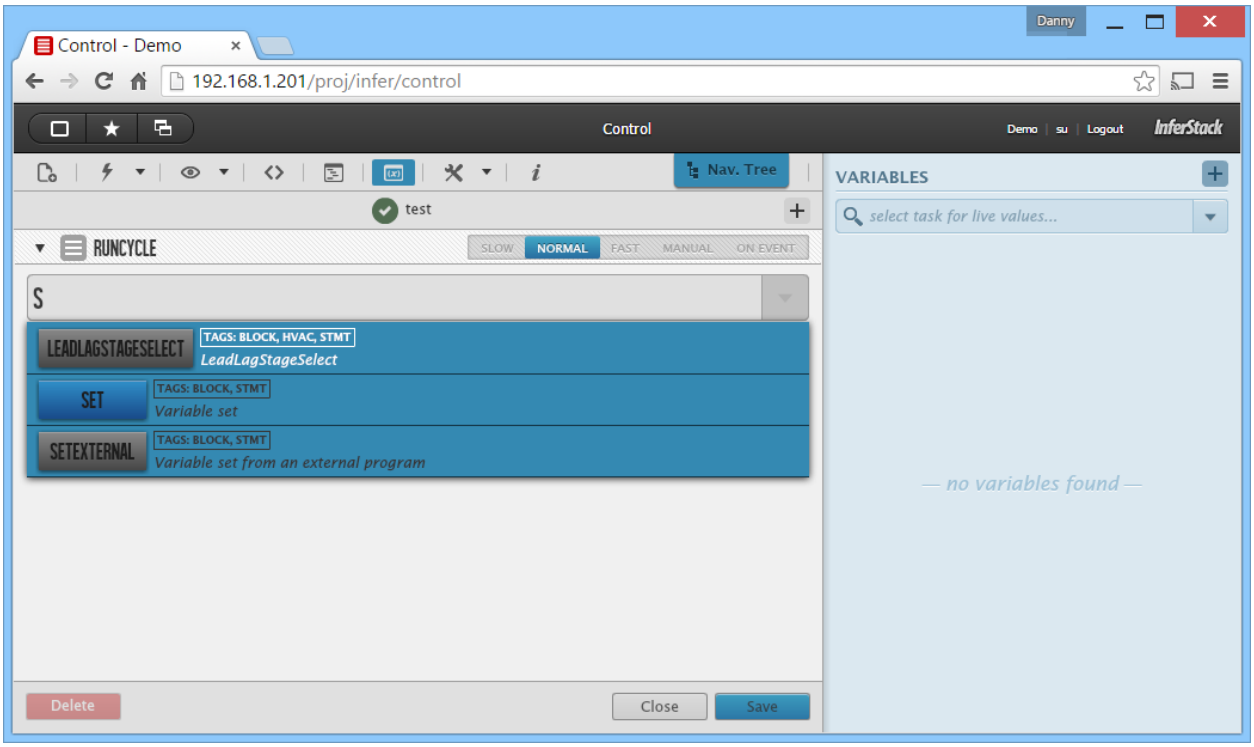
The easiest way to write bLine is to press space to see available blocks.



You can select a block …

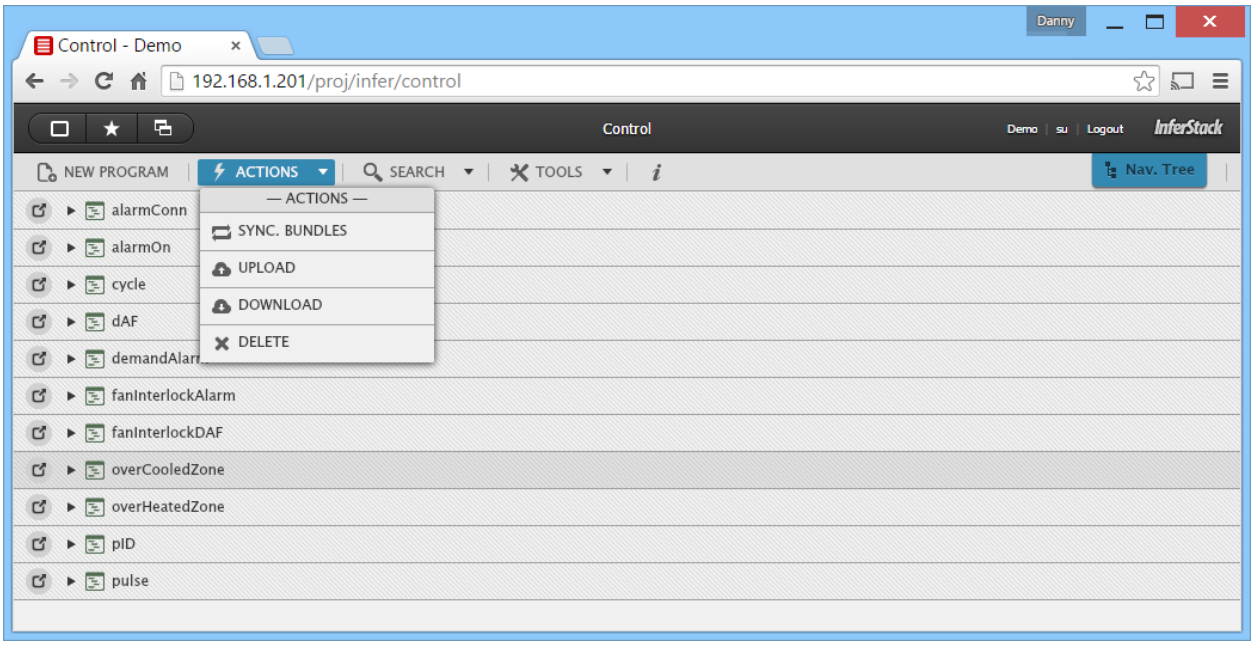


or as you type, the available blocks changes. Like pressing S.



Actions

Actions menu includes Sync Bundles, Upload, Download and Delete.



Sync Bundles

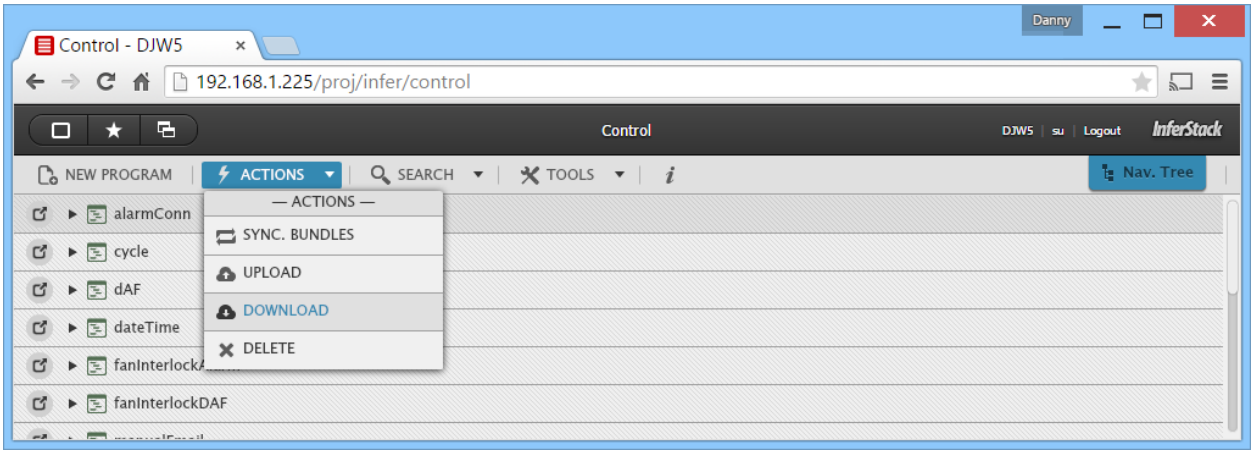
Sync Bundles allows you to synchronize bundles after changes.

Upload

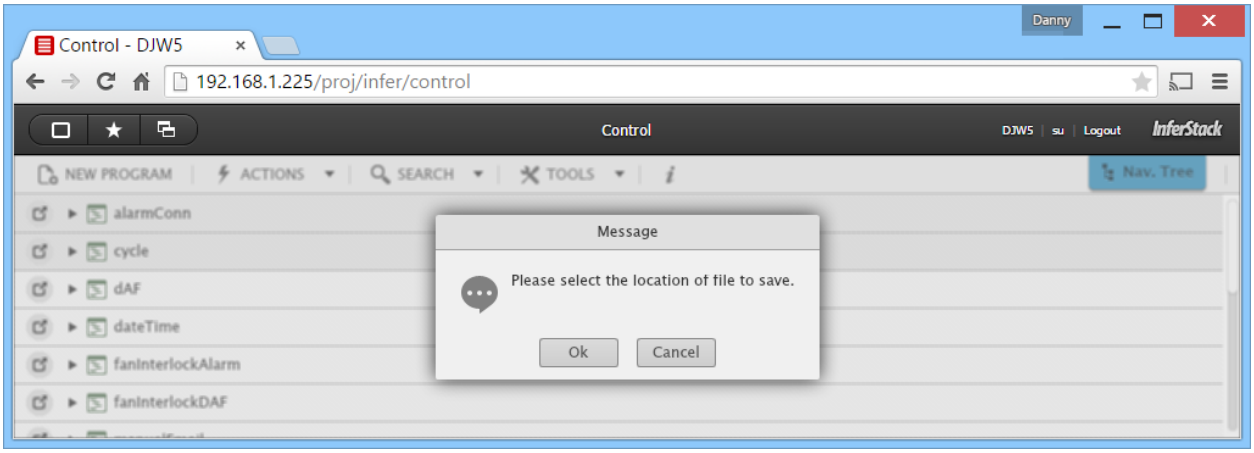
Upload program(s) allows you to up one or more programs.

Download

Download program(s) allows you to download one or more programs. Select one or more programs.



Select one or more programs and press Download.



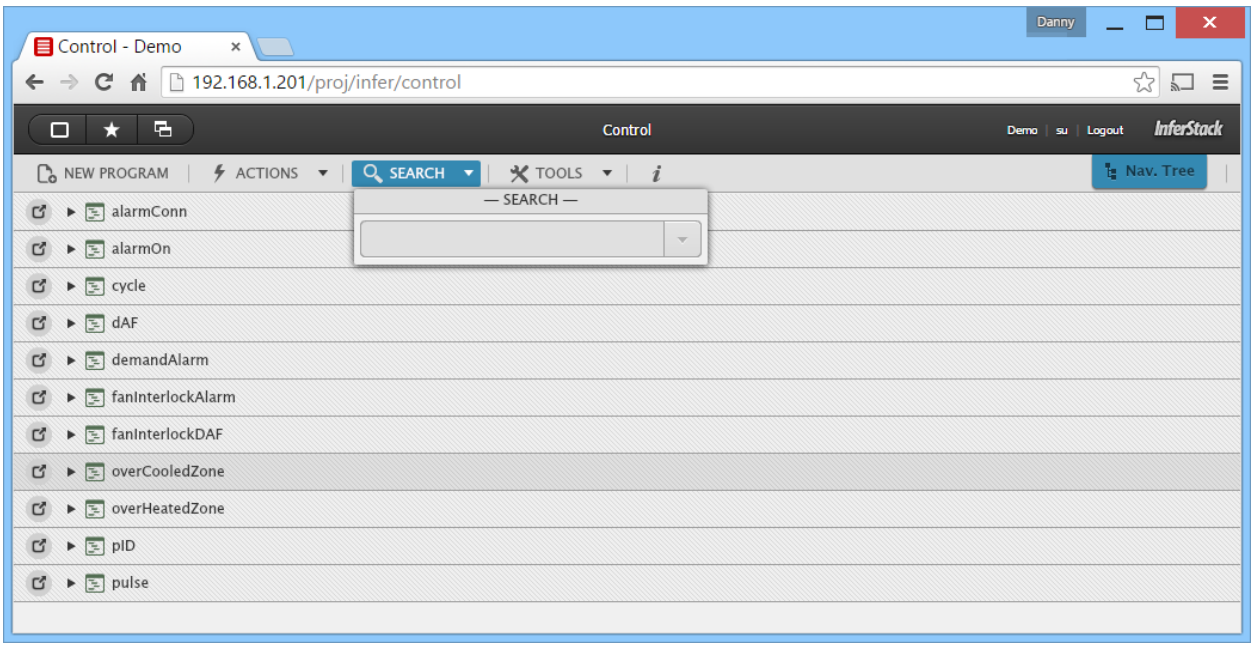
Press Ok and select the location for your bline file.

Delete

Select one or more programs and press Delete.

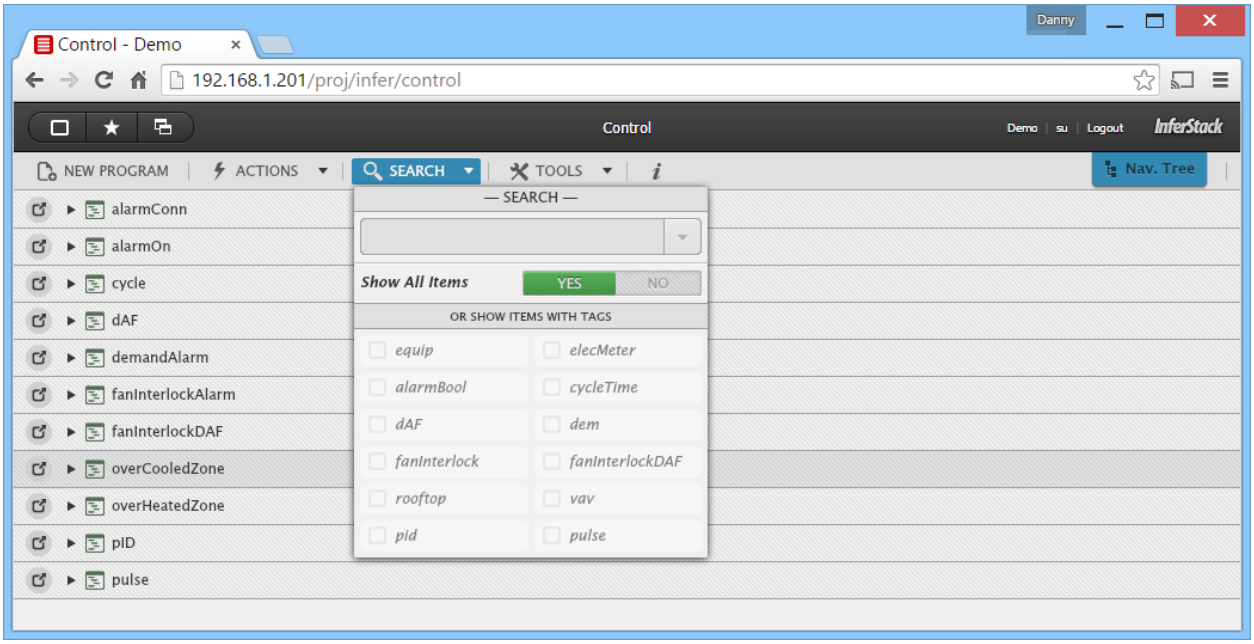
Search

Search menu provides a search box. In addition, the down arrow to the right provides Search Options.



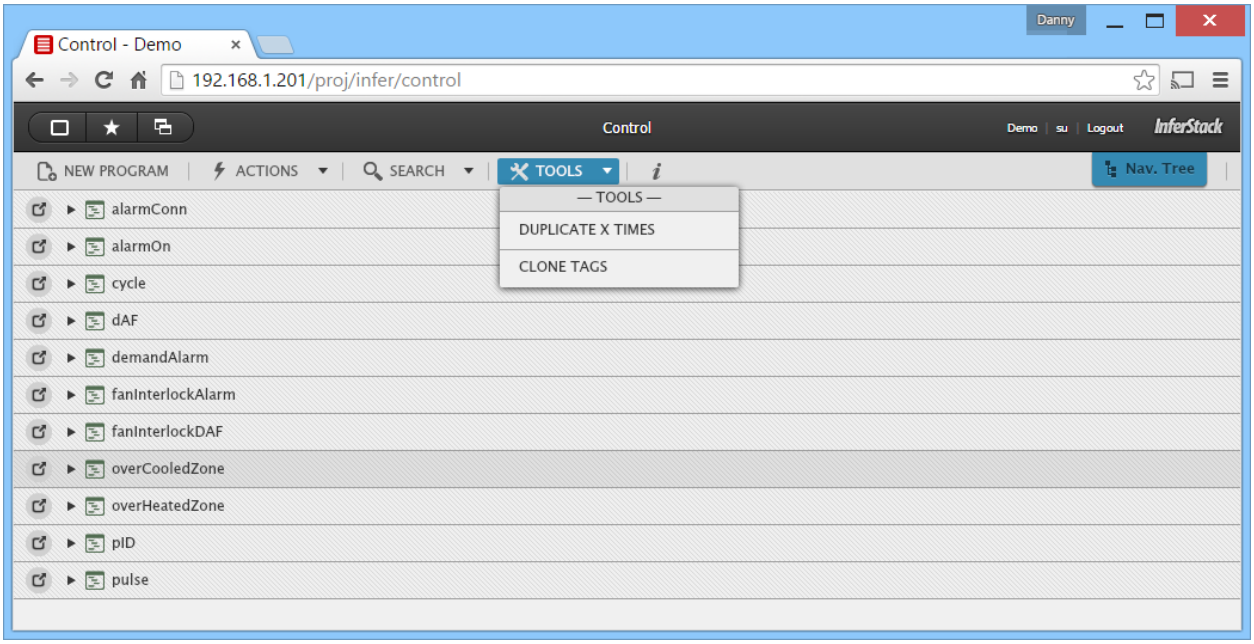
Search Options

Search Options allow you to choose whether to show all items or show items with tags.



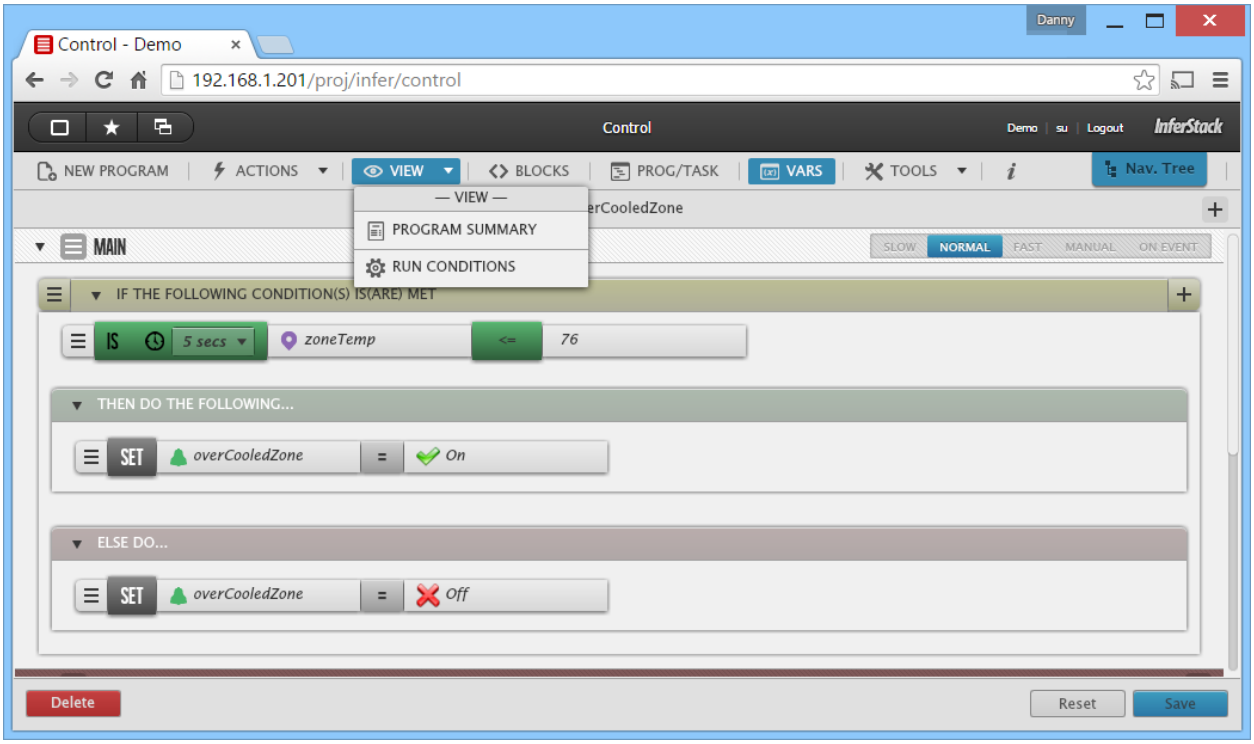
Tools

The Tools menu provides access to additional tools like Duplicate X Times and Clone Tags.



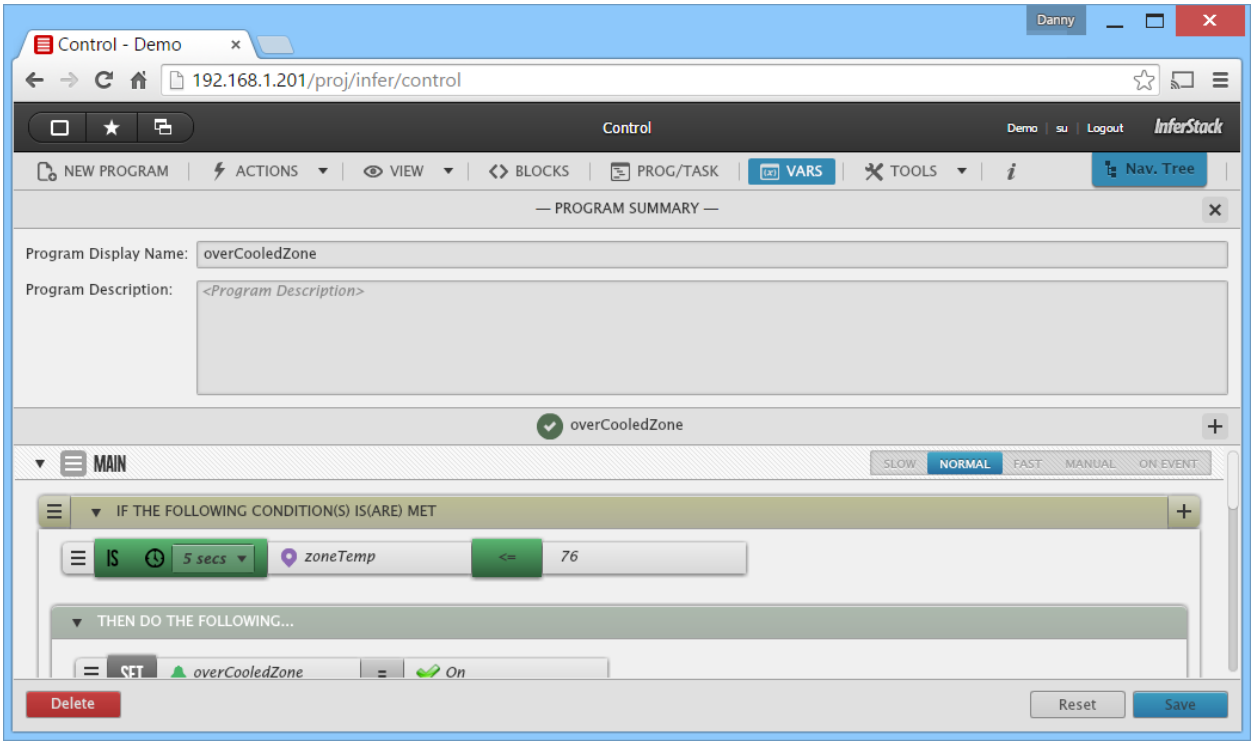
View – Program Summary, Run Conditions

Once a program is open, you will have a View Menu. The View Menu contains Program Summary and Run Conditions.



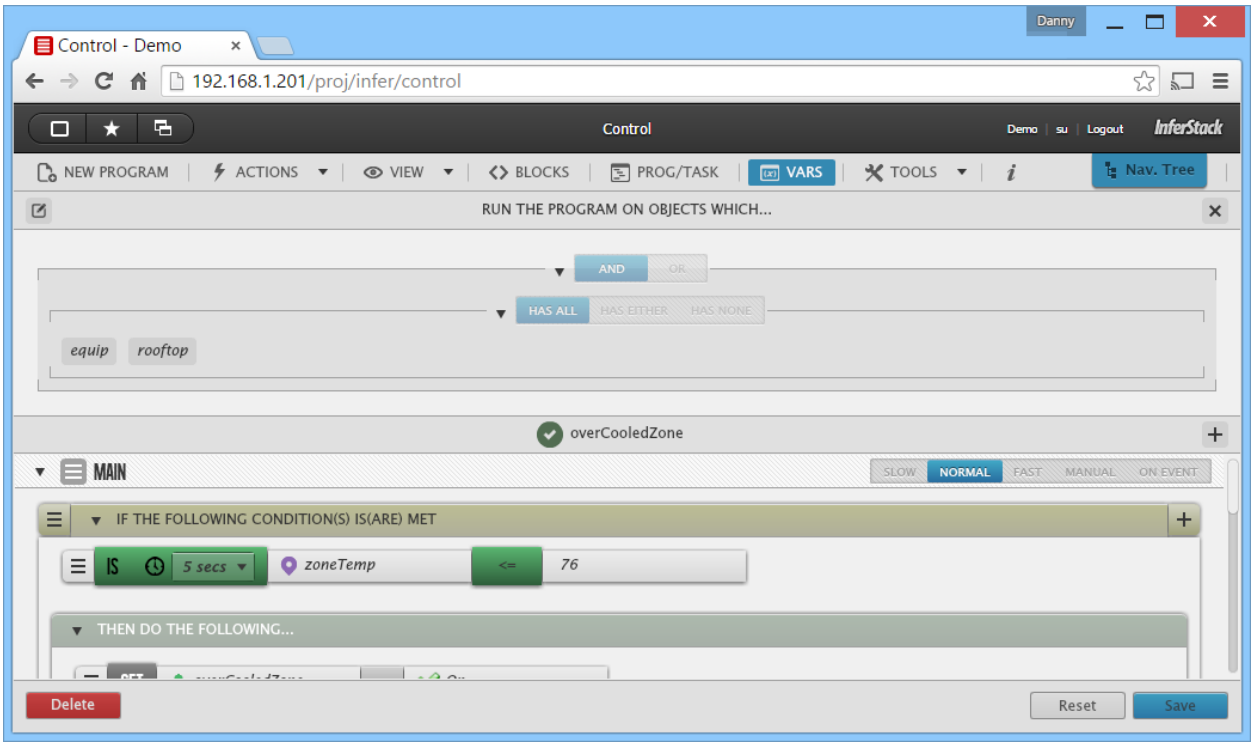
Program Summary

Program Summary provides access to name and description.



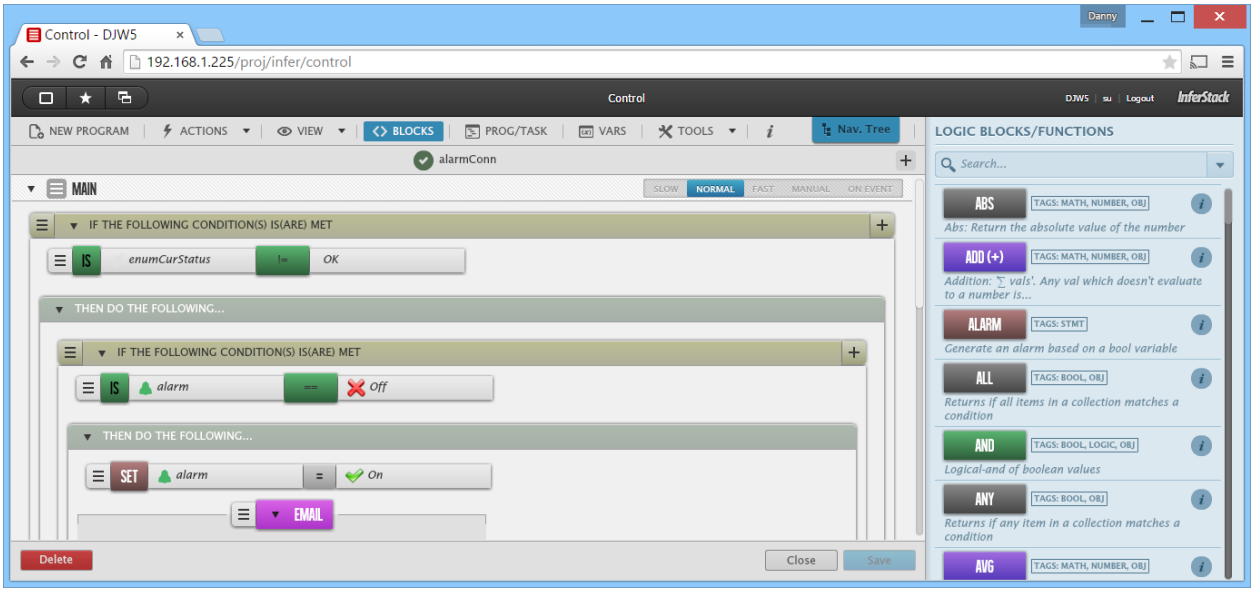
Run Conditions

Run conditions provides access to the tags that determine when the program will run. This program runs on equipment with a rooftop tag.



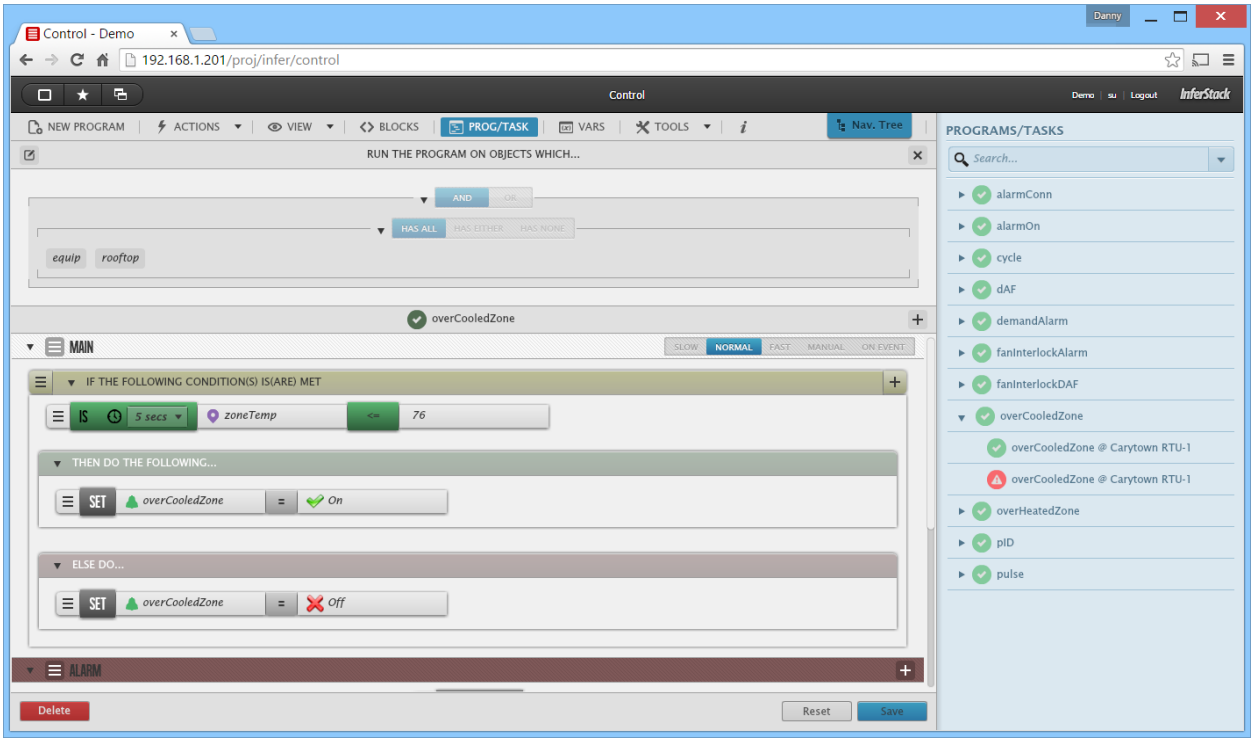
Blocks

The Blocks Menu causes the Logic Blocks to appear or disappear from the right sidebar when pressed. This allows you to review all of the available blocks and drag them into your program. The blocks are listed alphabetically and colored to make them easy to recognize. Most of the Booleans are green, Numerics are purple and Strings are grey.

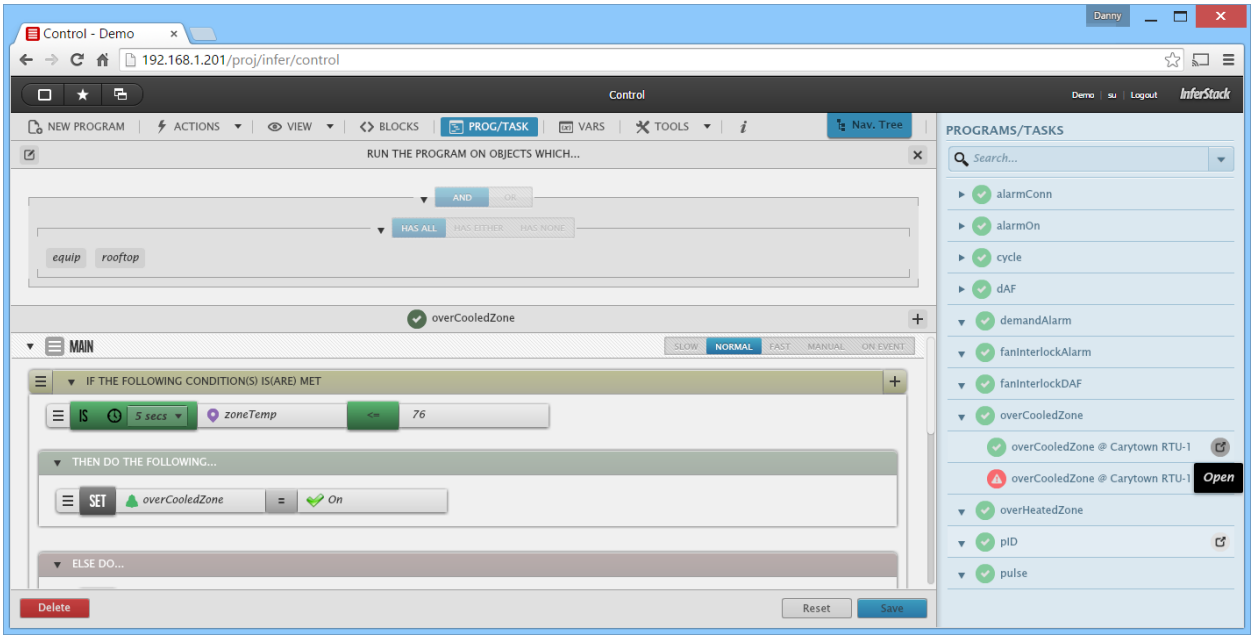


Prog/Task

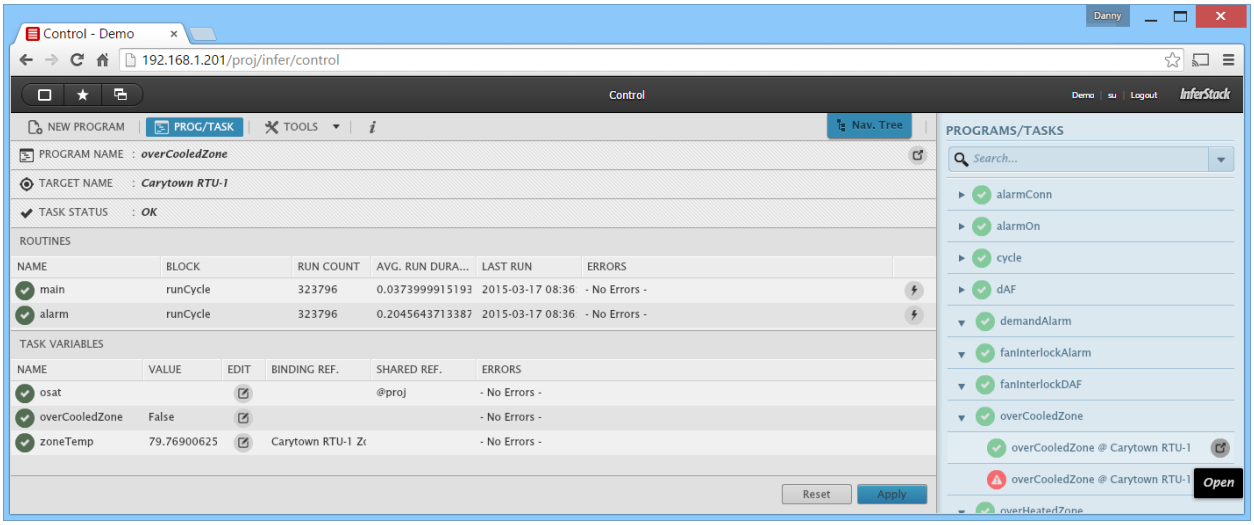
The Prog/Task Menu causes the Programs/Tasks to appear or disappear from the right sidebar when pressed.



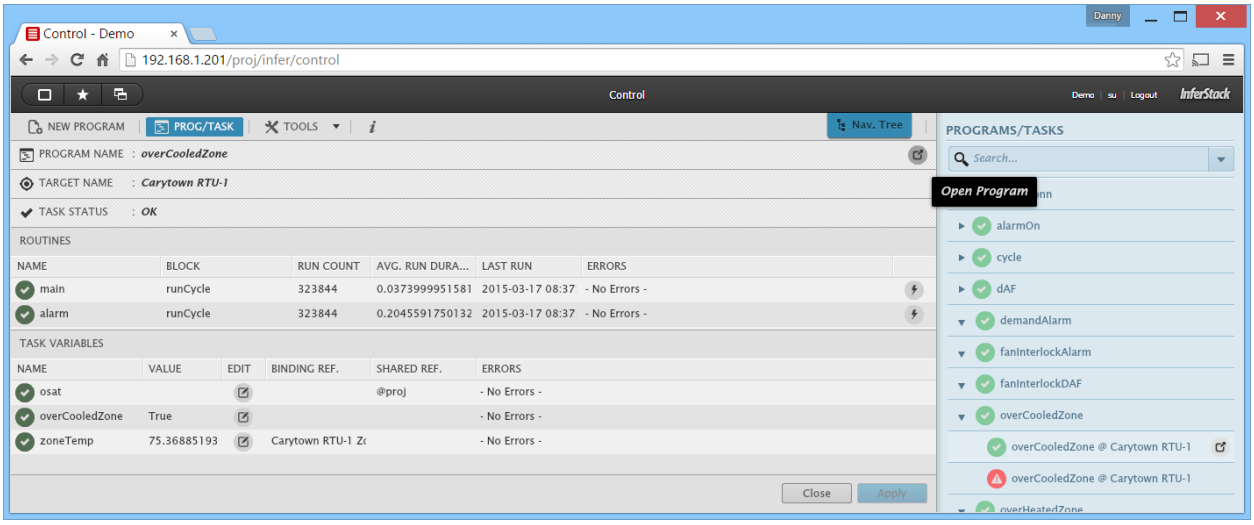
Press the right arrow to see the individual tasks. Press open:



To see the details:

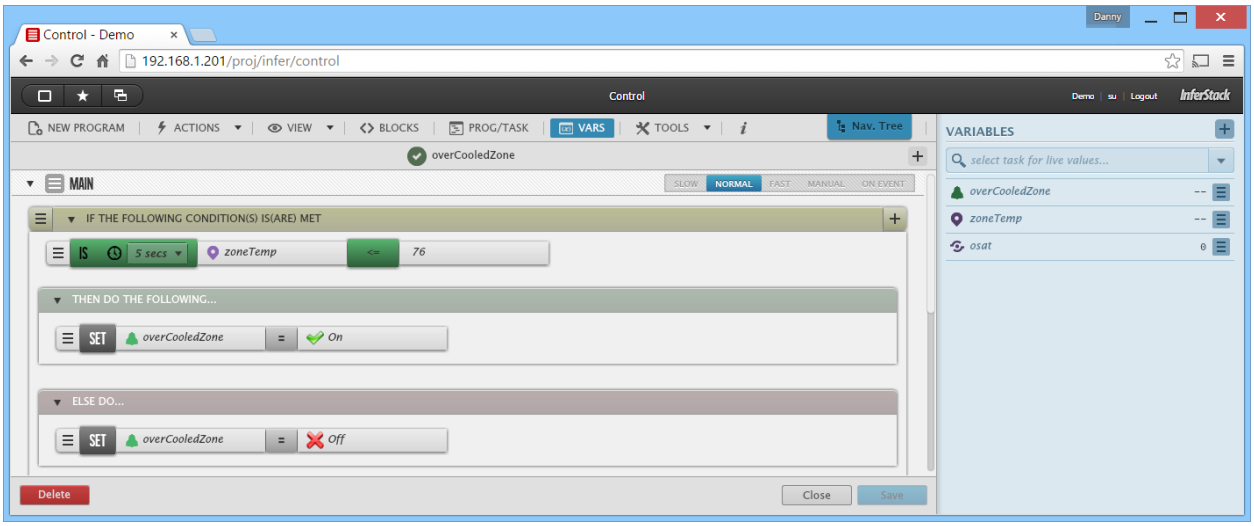


Press Open Program to return to the program.



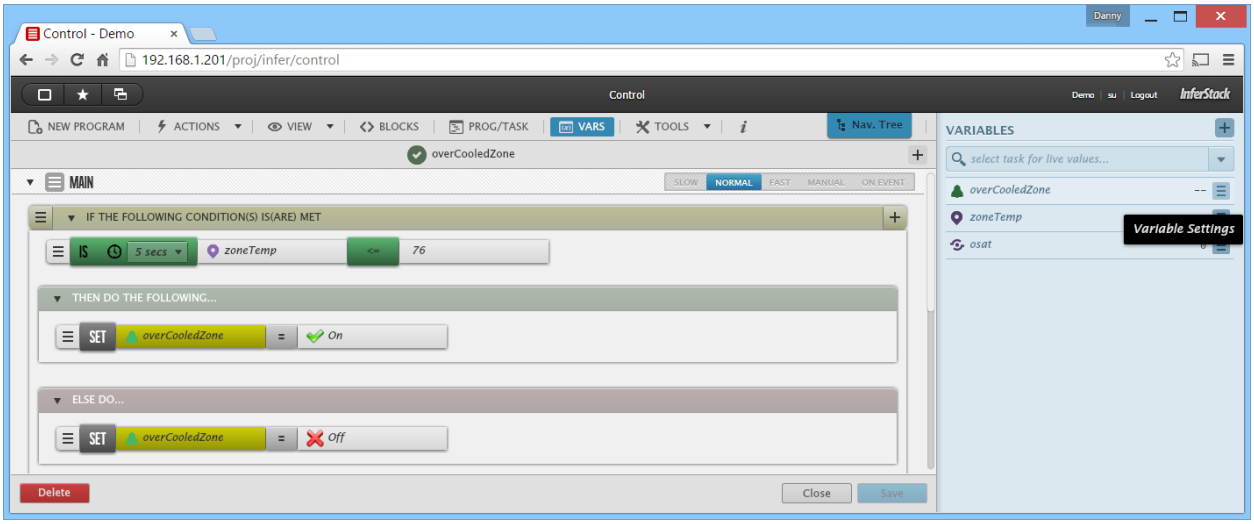
Vars

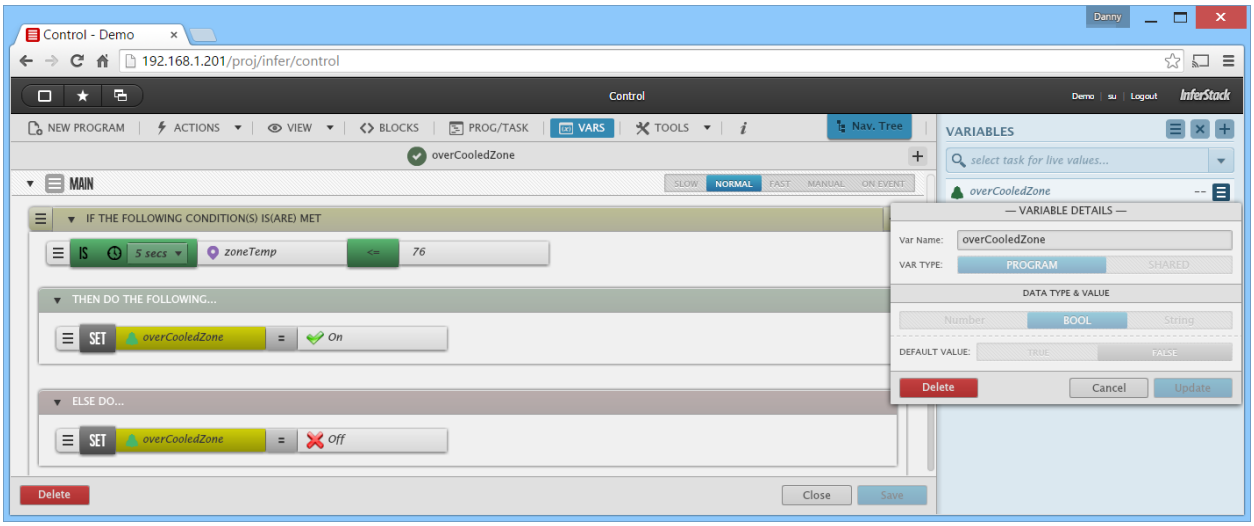
The Vars Menu causes the Variables to appear or disappear from the right sidebar when pressed.



Variable Settings

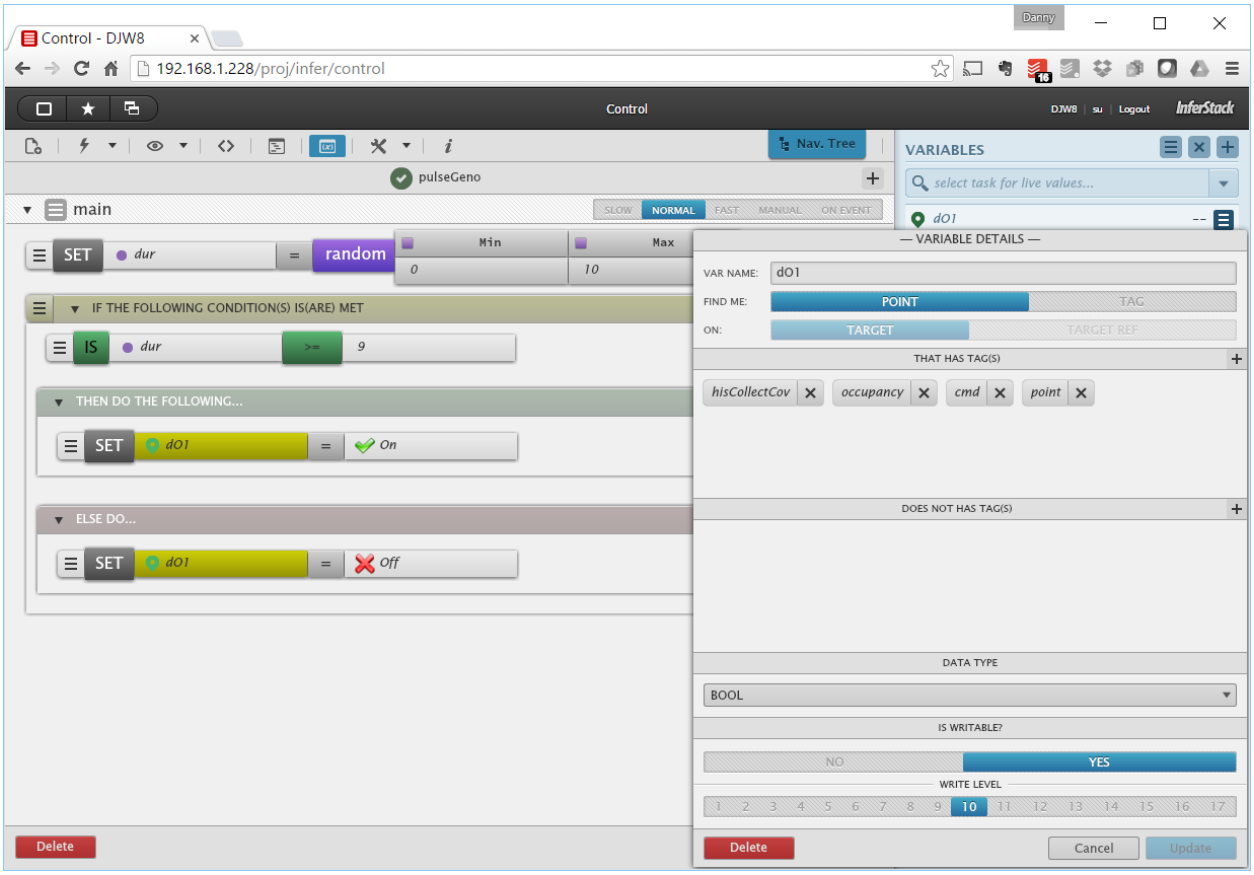
Press Variable Settings to see variable details.





Write Level

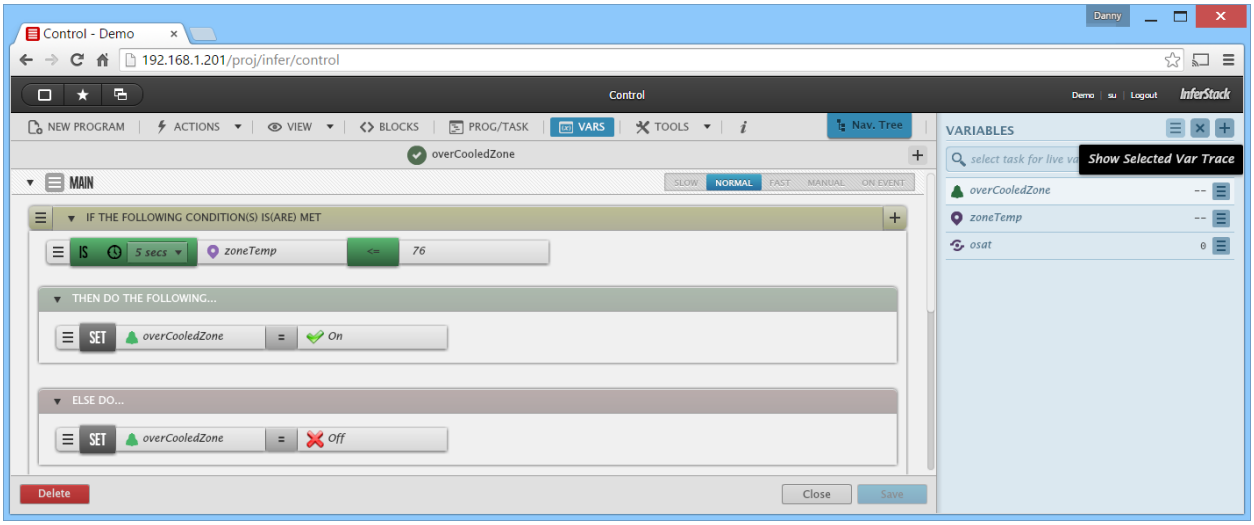
From Variable Details, you can edit properties of the variable including changing the write level for writeable variables.

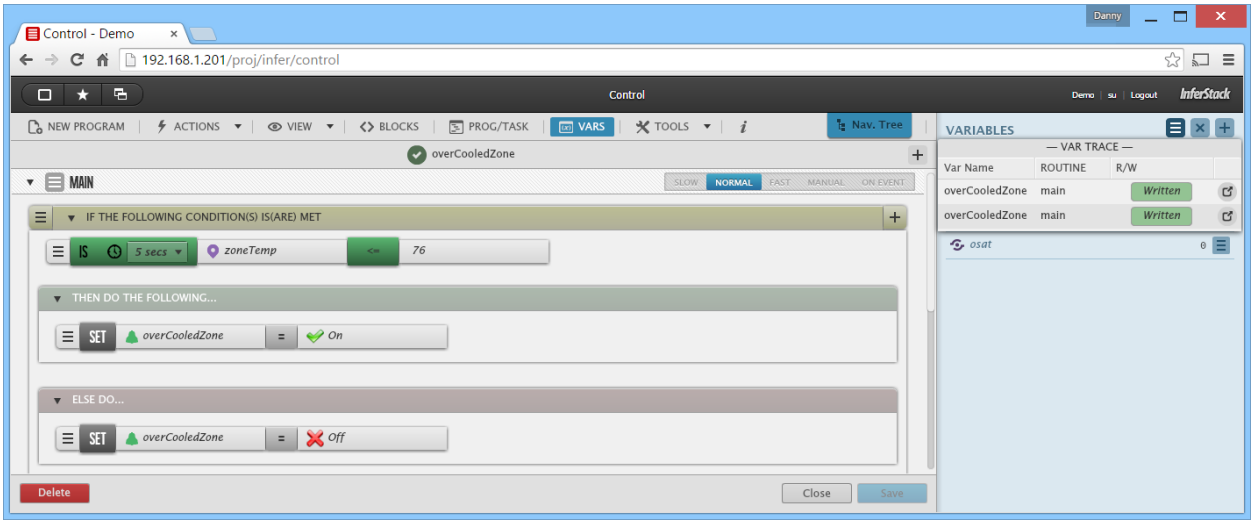


Once you select one or more variables, you will see three menu items: Show Selected Variable Trace, Delete Selected Variable(s) and New Variable.

Show Selected Variable Trace

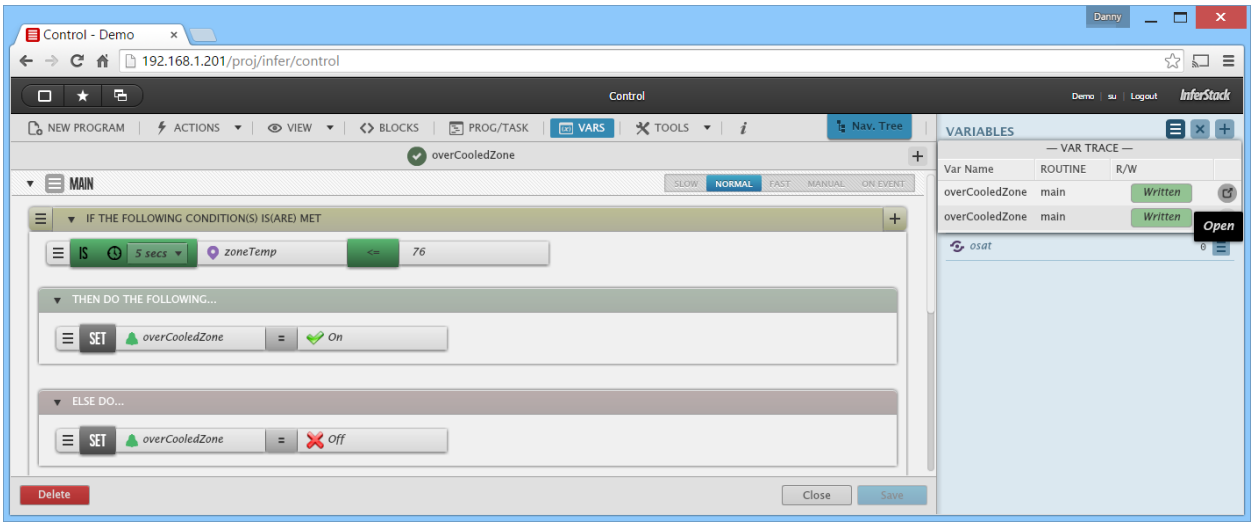
You can review the variable trace.





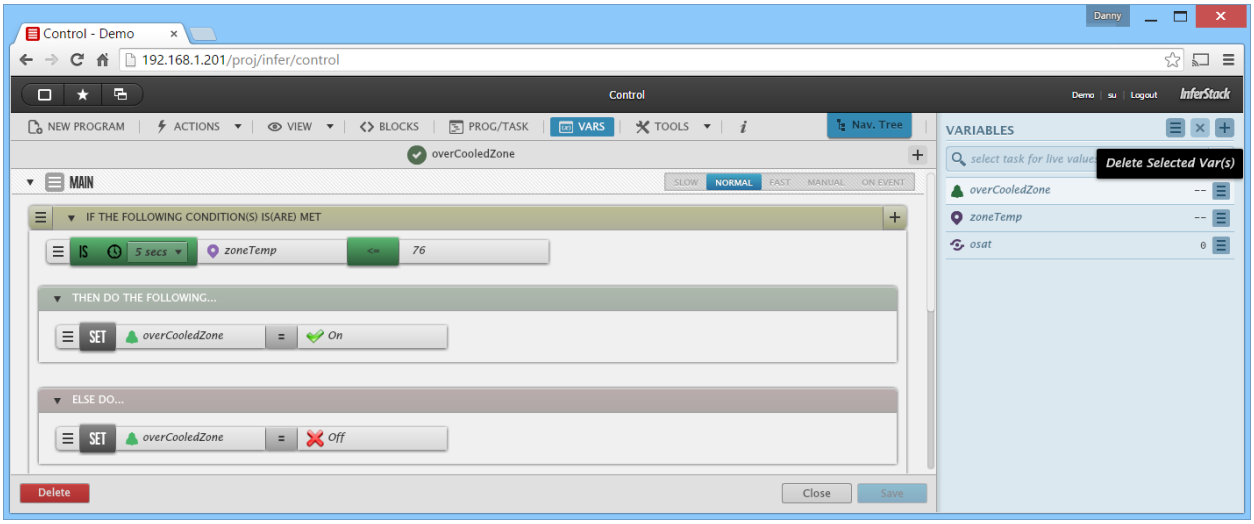
Open

Open the program, if not already open.



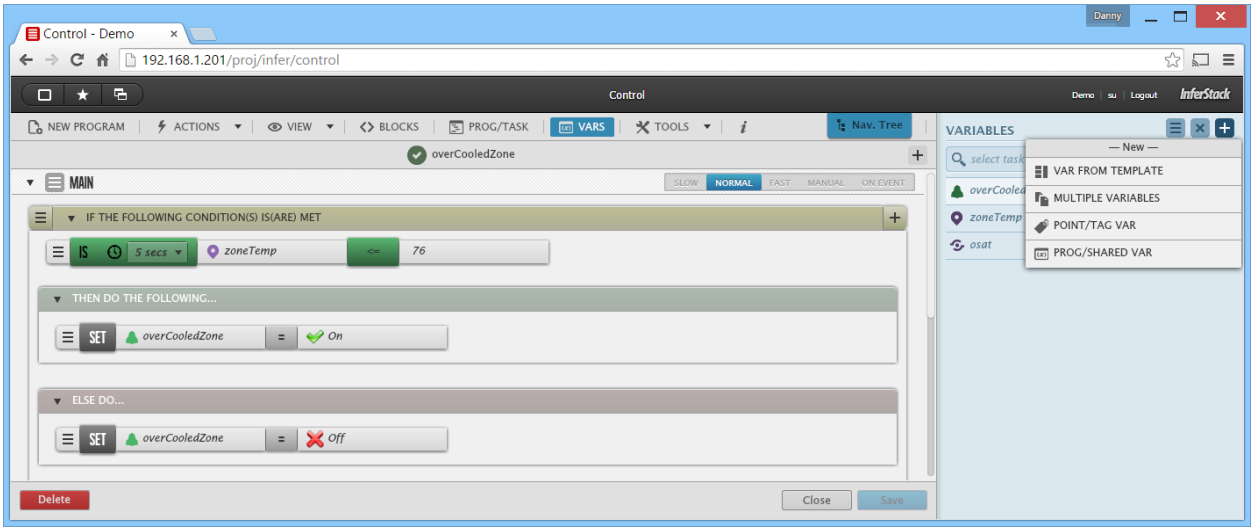
Delete Selected Variable(s)

Delete the selected variable(s).



New Variable.

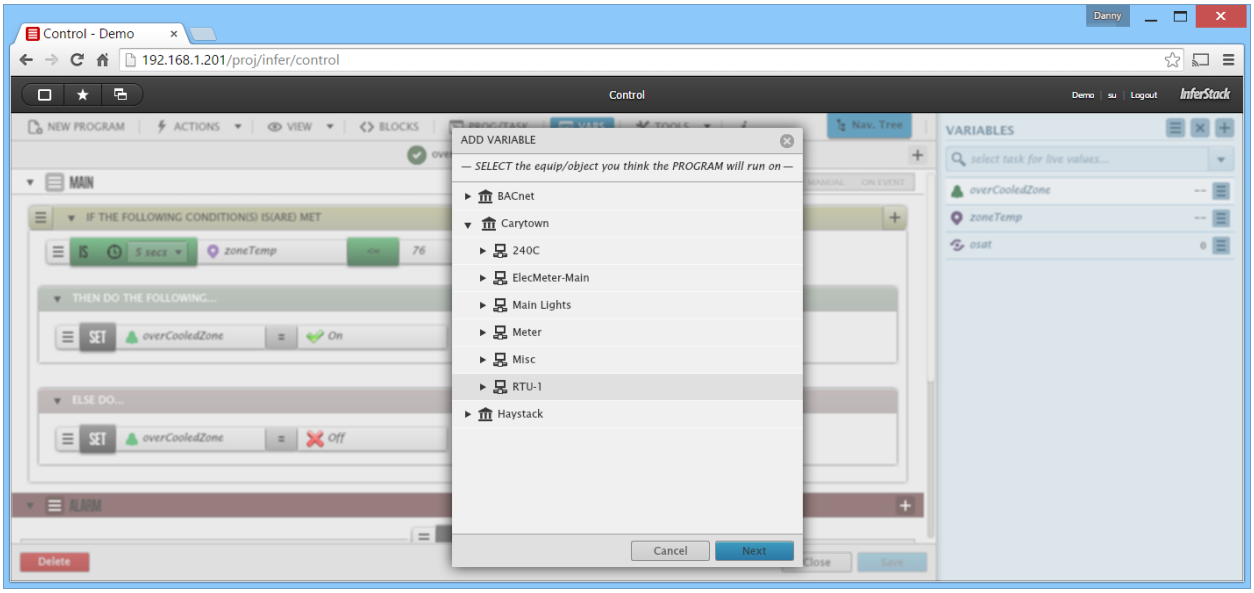
Press the plus (+) sign in the upper right to add a new variable. You may choose Var From Template, Multiple Variables, Point/Tag Var or Prog/Shared Var.



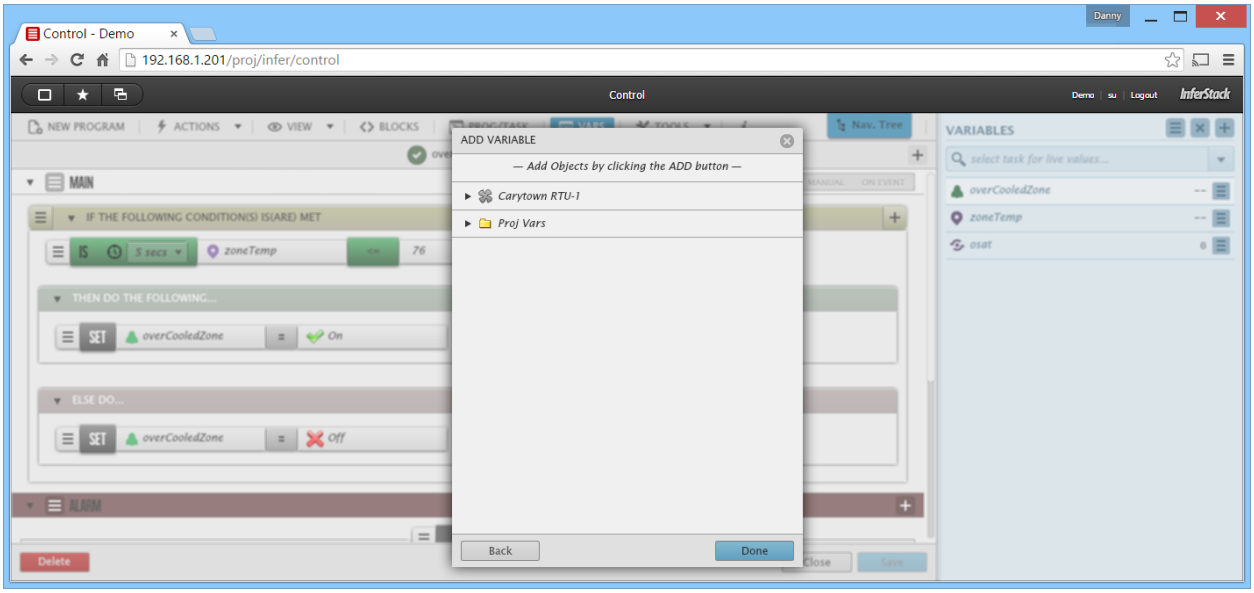
VAR FROM TEMPLATE

Select var from template and select an object you think the program will run on.

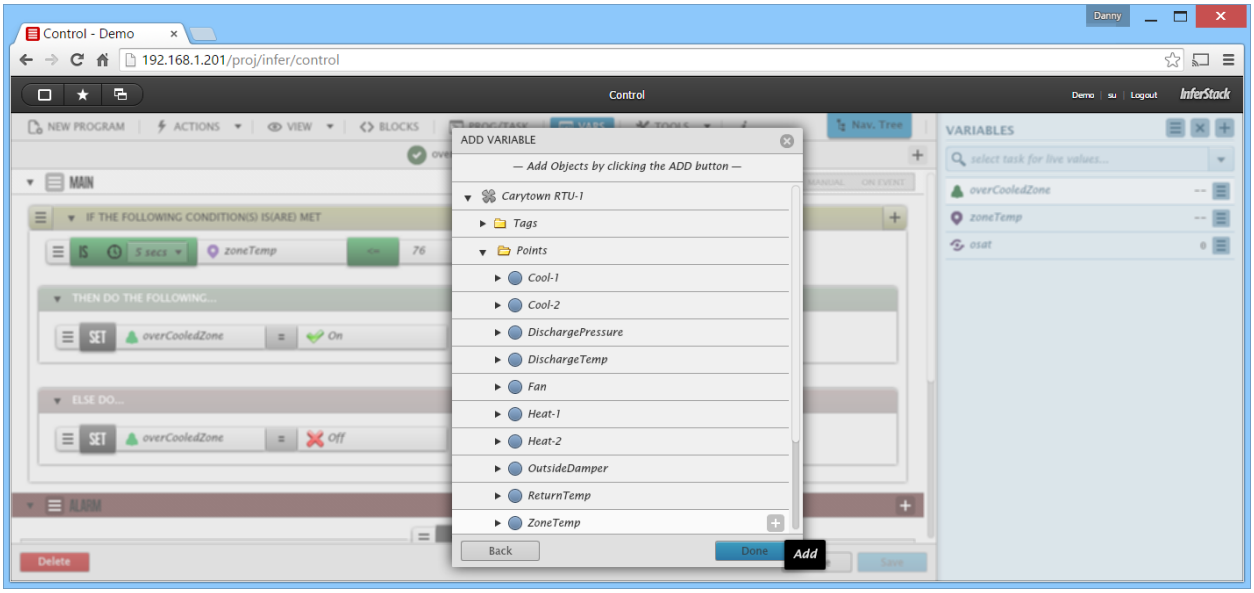
Note: Remember this should be equipment.



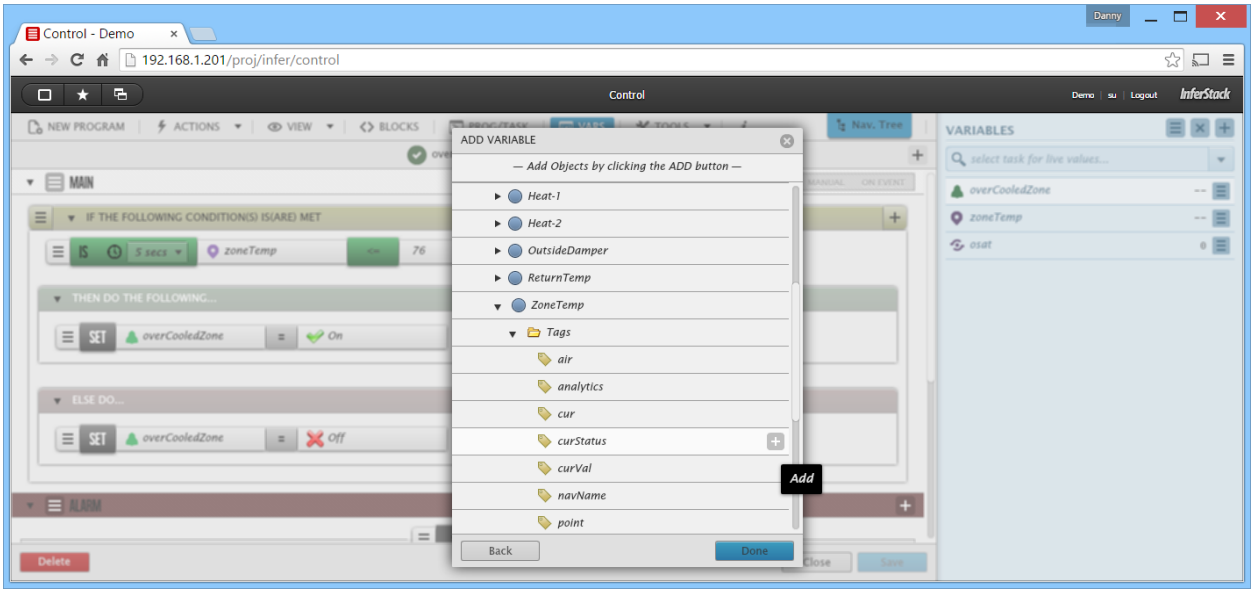
Press Next



Expand the tree to see objects available to add. These typically include Tags, Points and Refs. The Points provide the points in the equipment and their tags (like curStatus). The Refs are also useful to get to site tags for example.

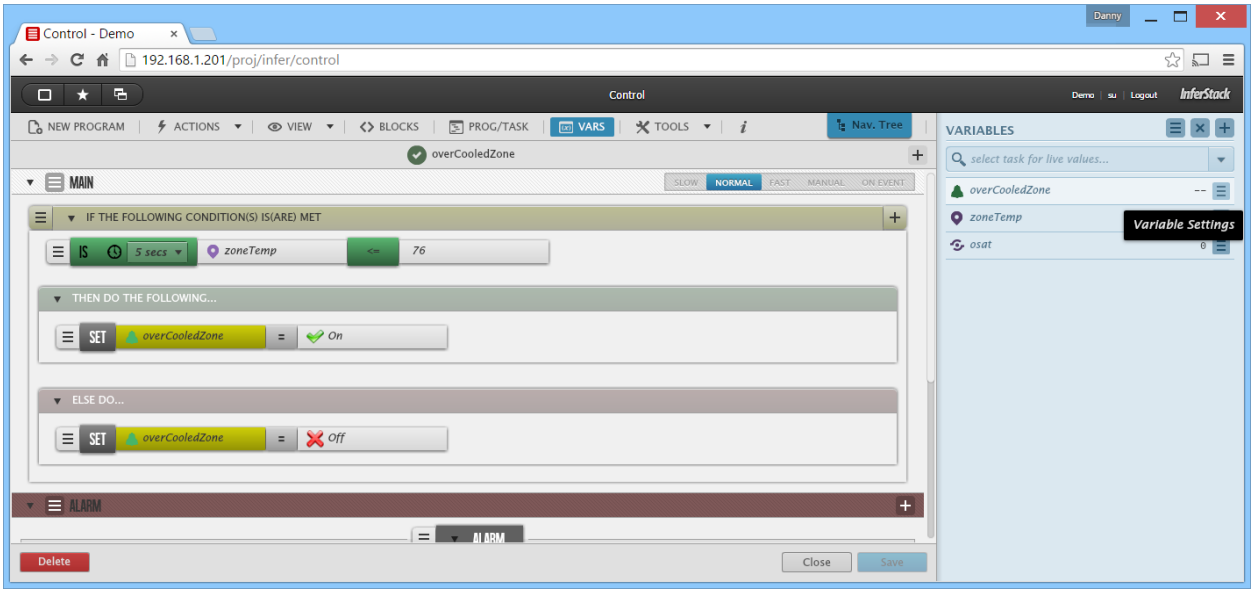


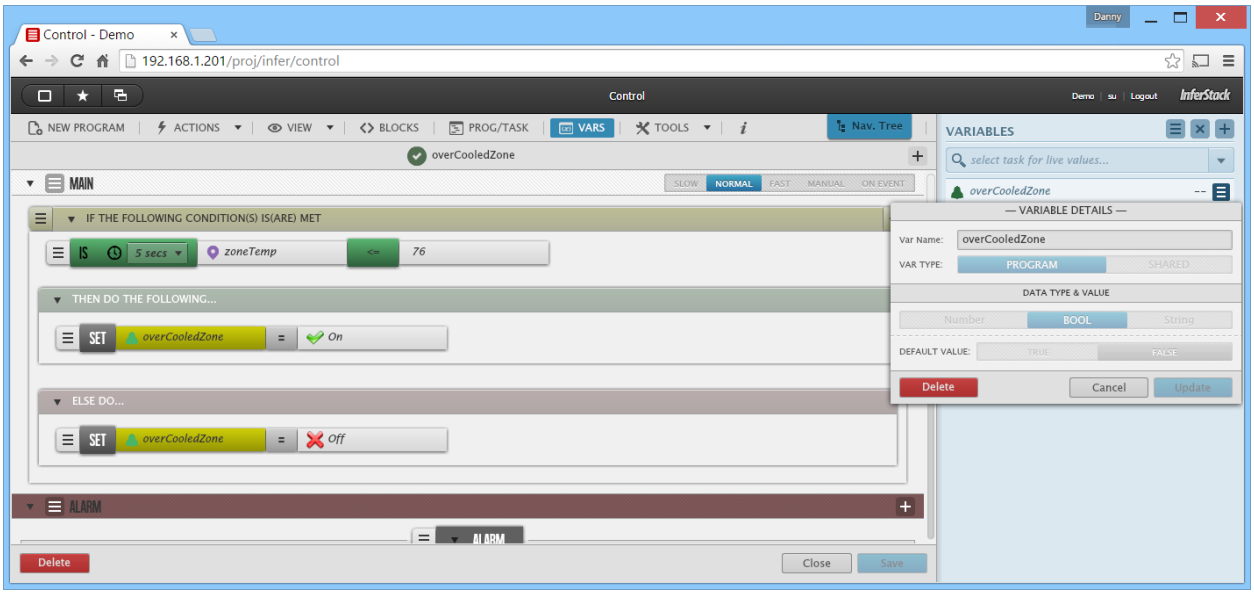
Add Objects by clicking the Add (+) button that appears when you hover over it.
You can also select tags on a point by expanding it.



Press Done when you have added the variables you need.

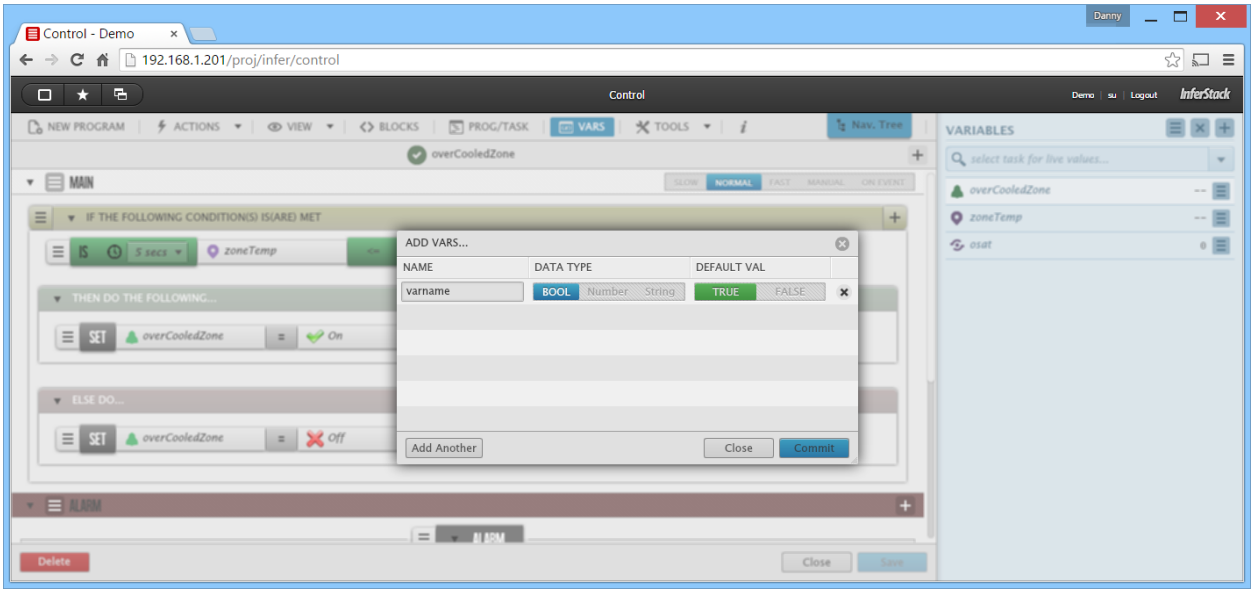
Press Variable Settings to see the details of any variable.





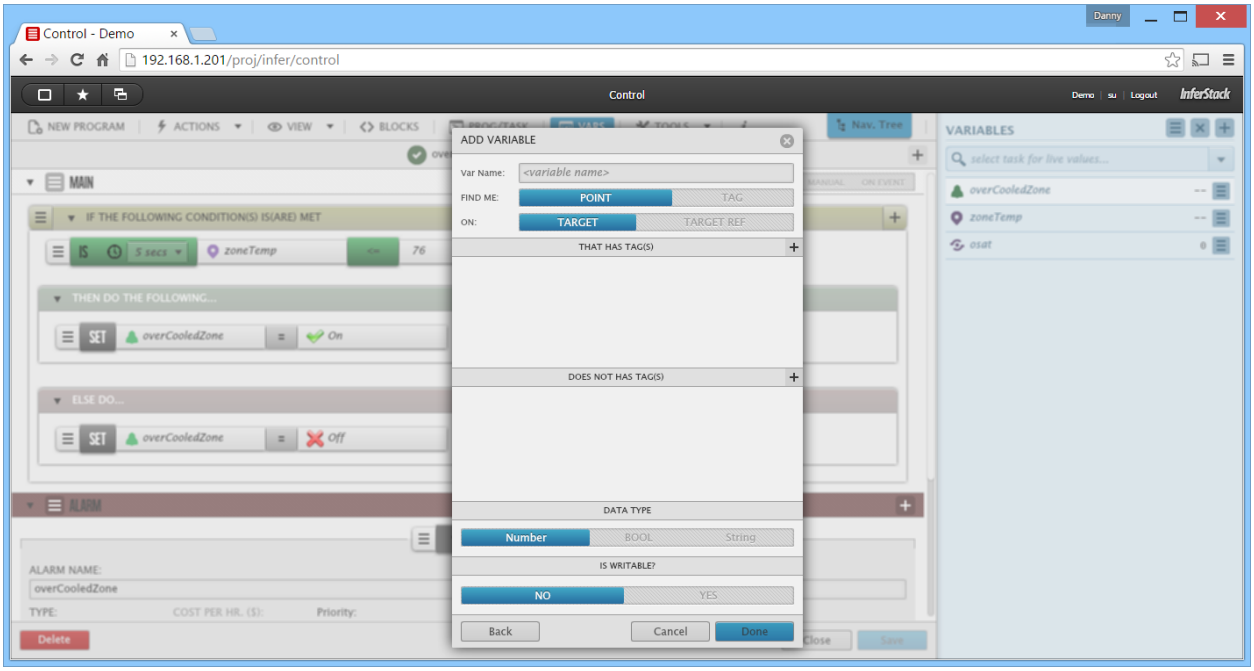
MULTIPLE VARS

Press Multiple Vars to create multiple variables. Type a name, select data type and default value, and press Add Another as many times as needed. Press Commit when complete.

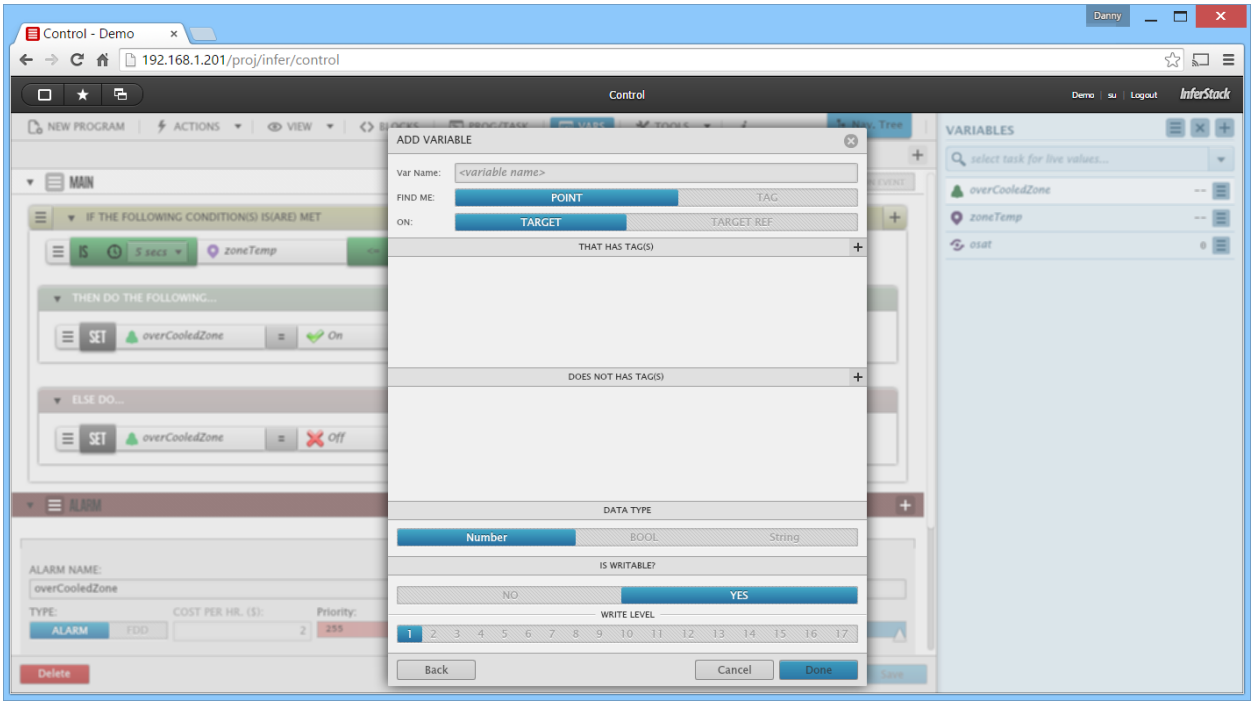


POINT/TAG VAR

Select Point/Tag Var and enter variable name (like curVal) and Data Type.



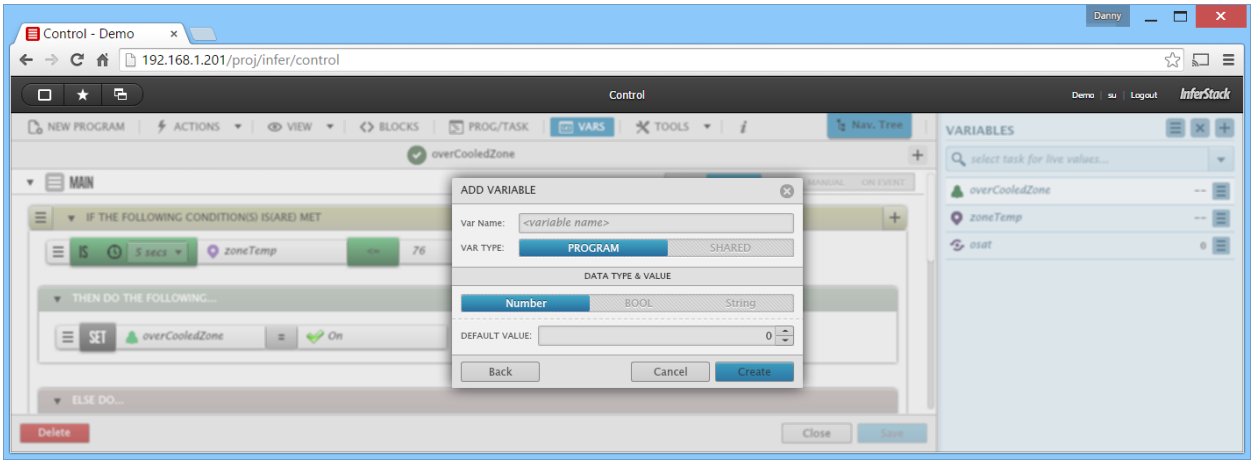
Check Is Writable to Set and choose Write Level.



.

PROG/SHARED VAR

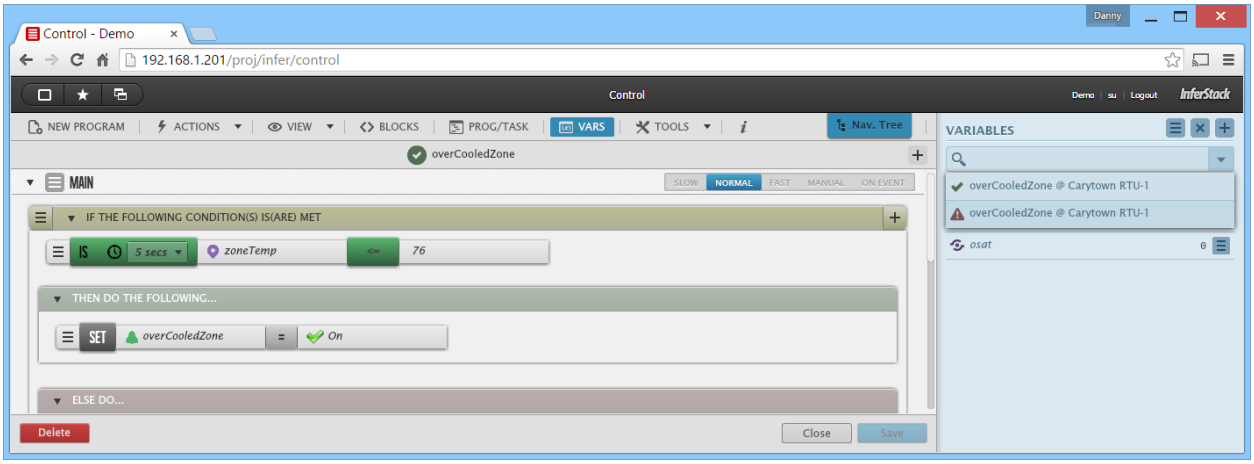
Select Prog/Shared Var



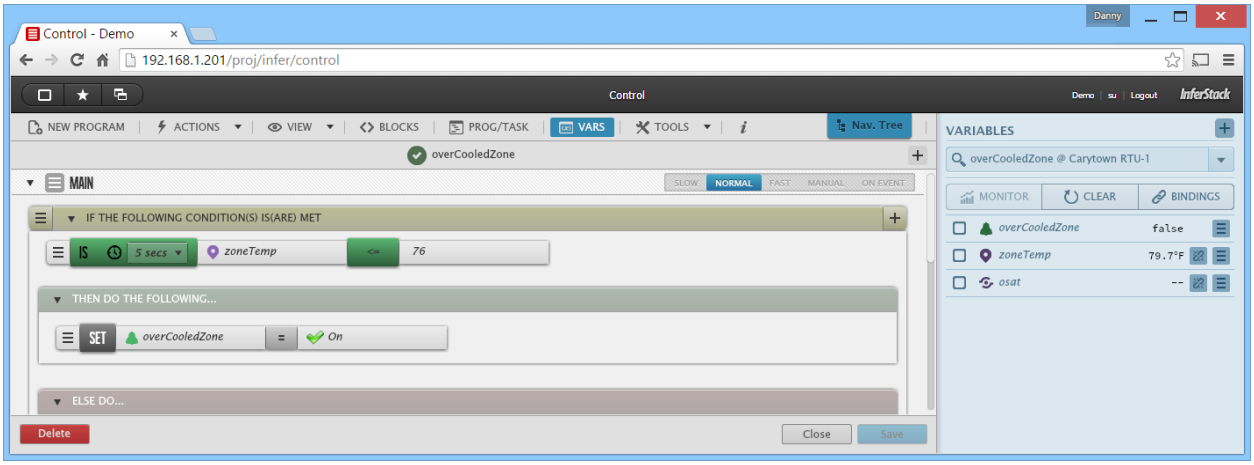
Name the variable, select Program or Shared, the datatype and a default value if desired and press Create.

Select Task For Live Values

Select the down arrow to see the tasks.



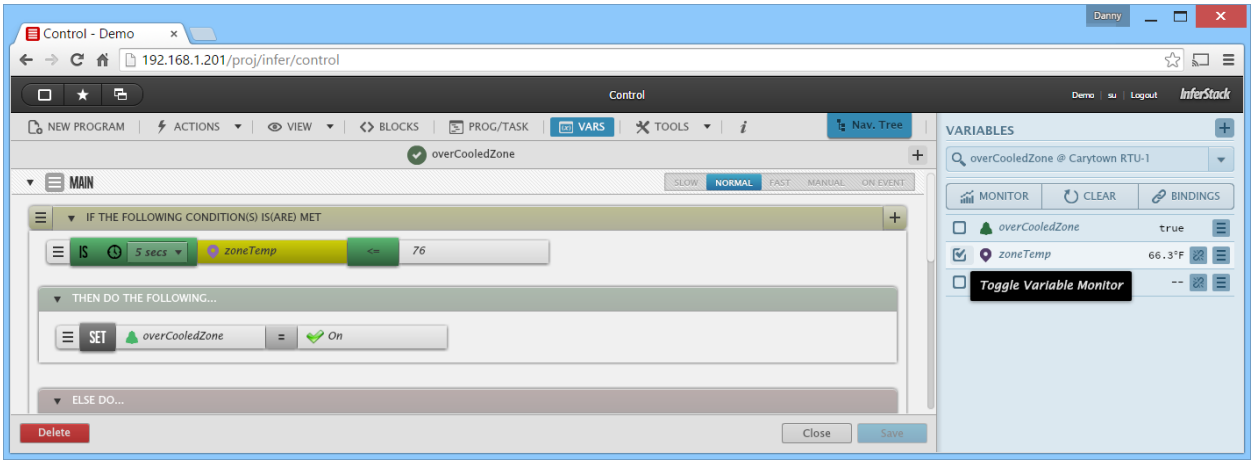
Select one to see live data.



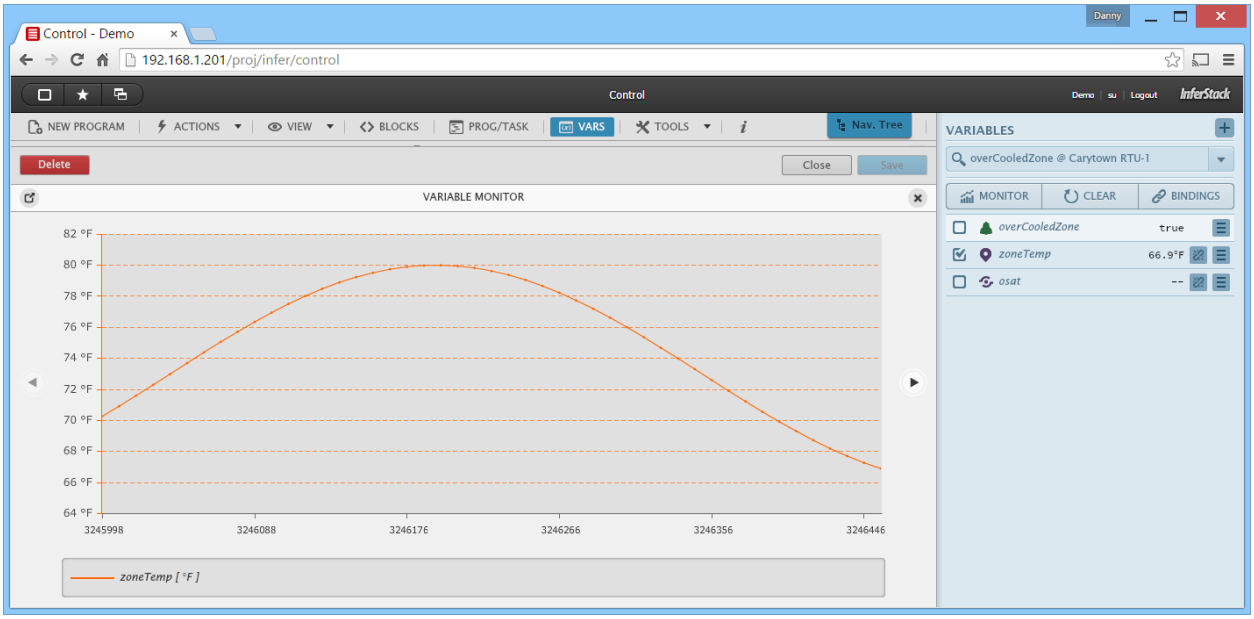
You will see Monitor, Clear and Bindings.

Monitor

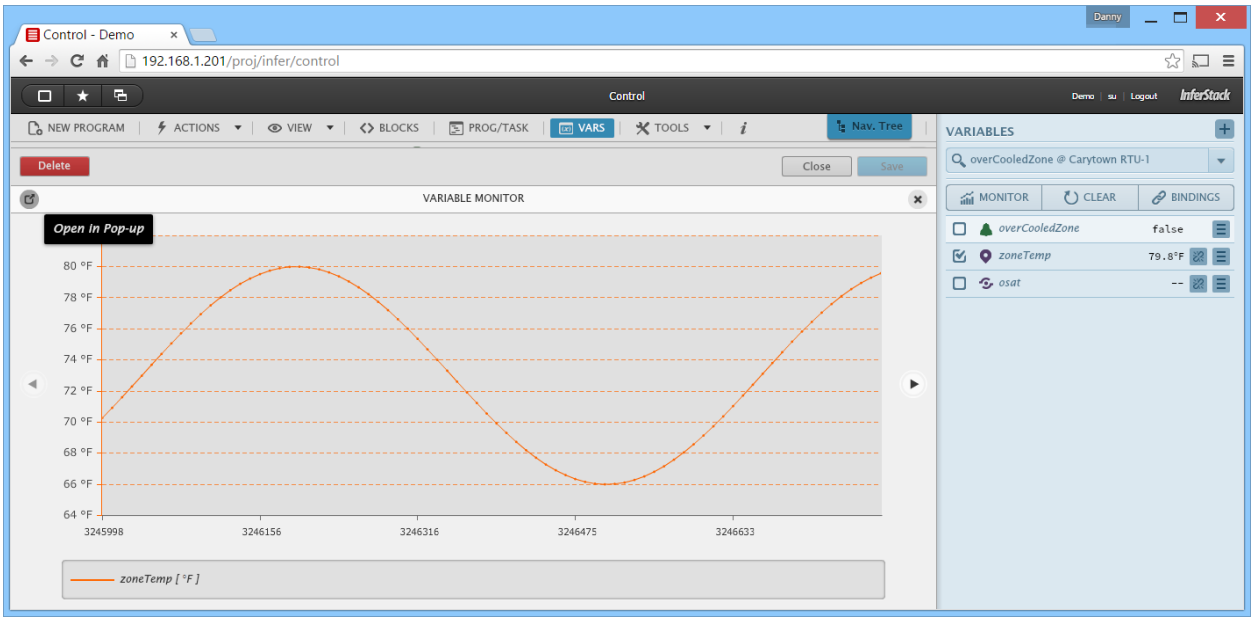
Press the check box beside a variable to enable the monitor.

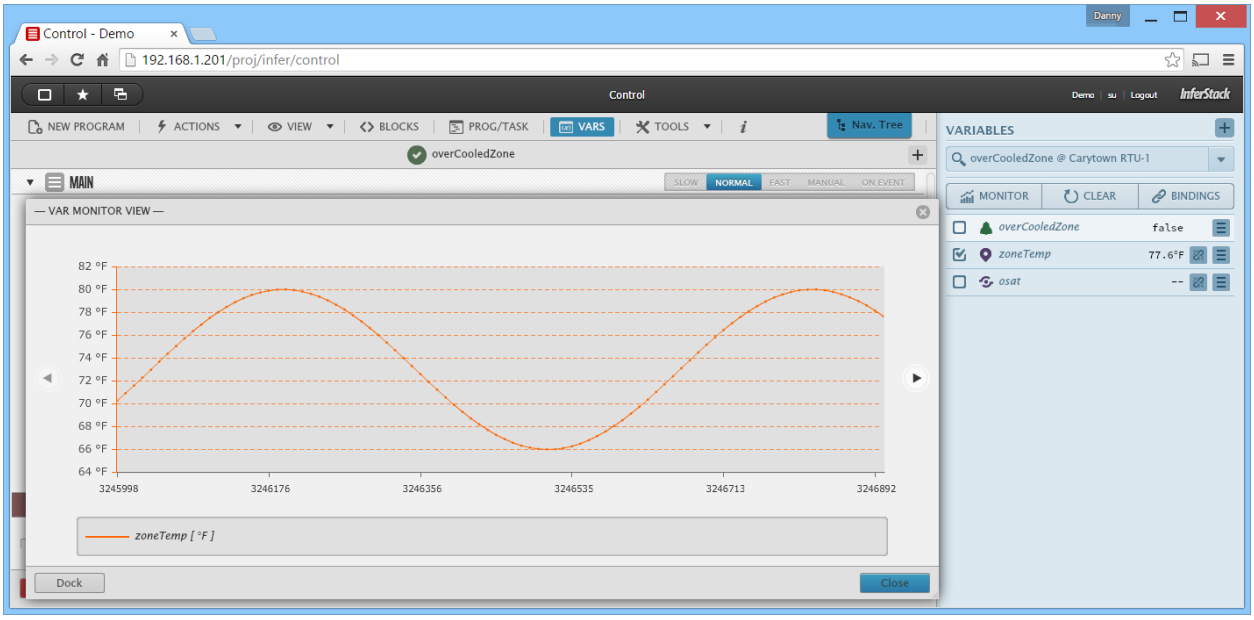


Press monitor.



Open in Popup

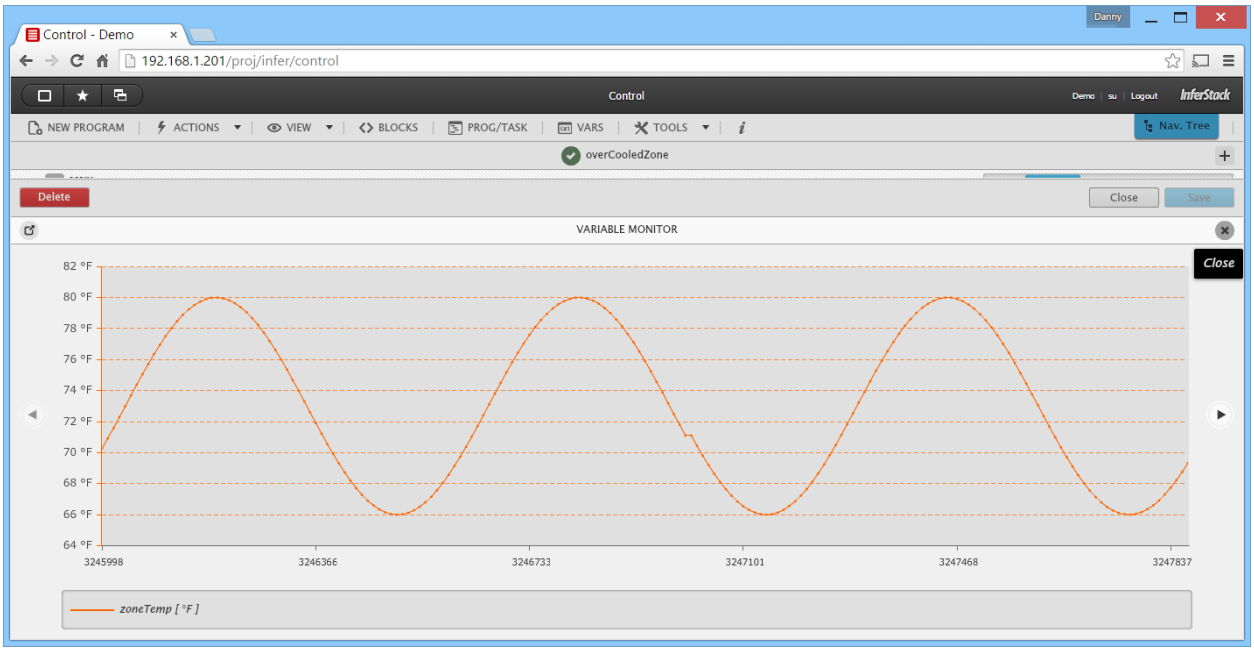




You can press Dock in the lower right to return to docked view.

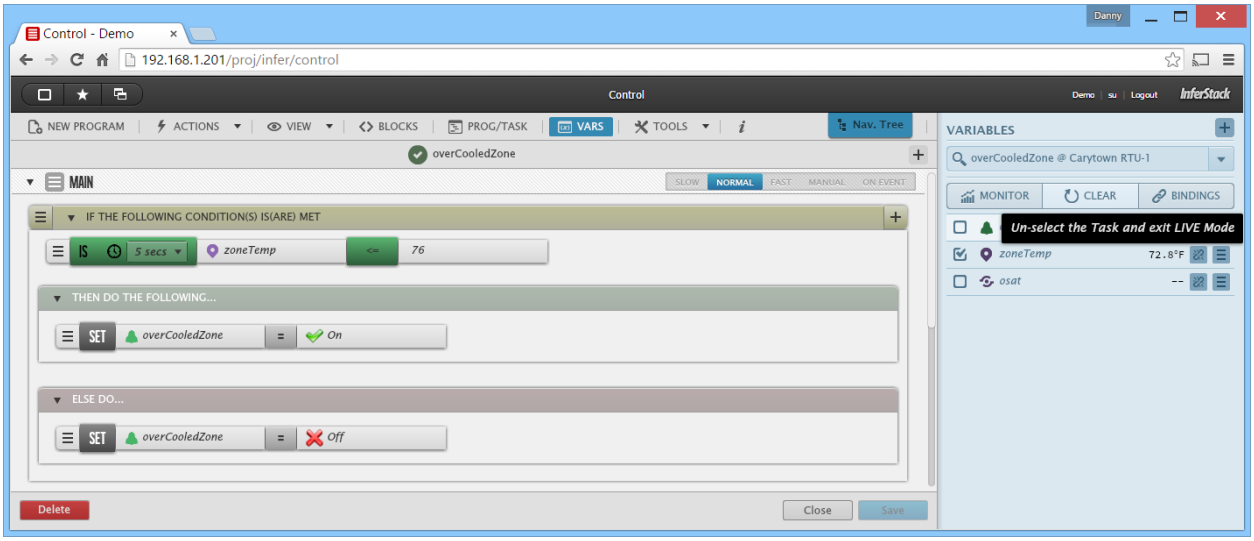
Close

Press Close to exit monitor (x icon, not Close program).



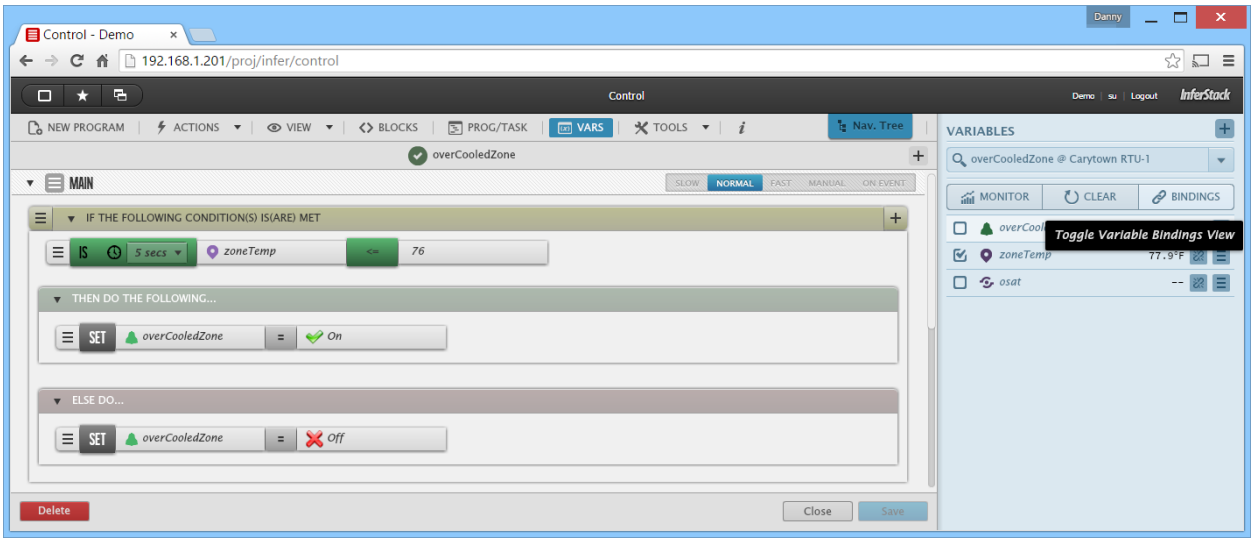
Clear

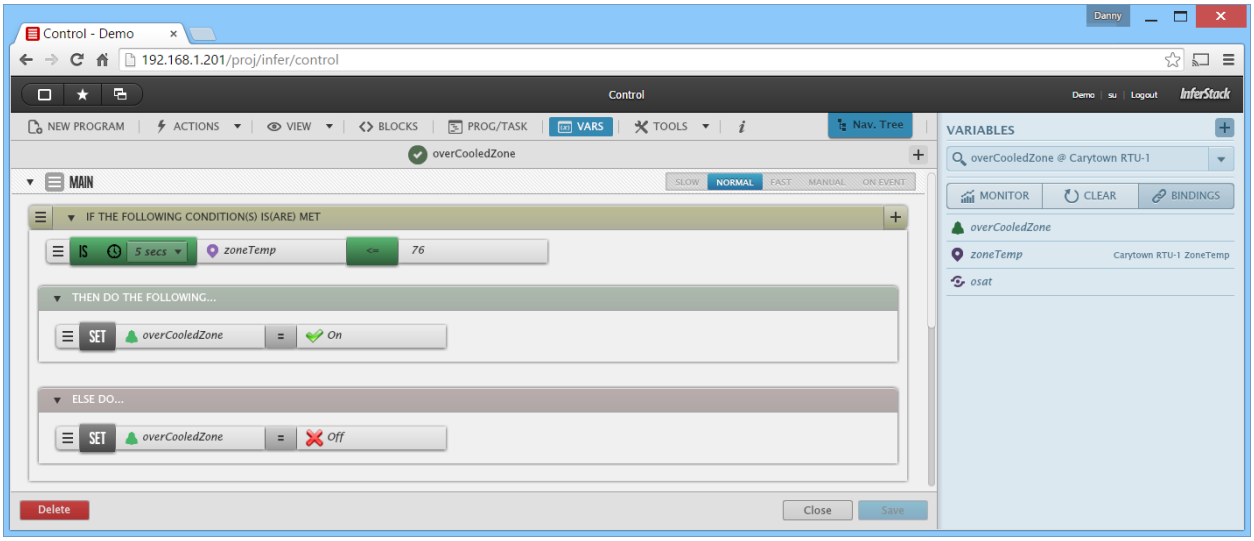
Press Clear to unselect the task and exit live mode.

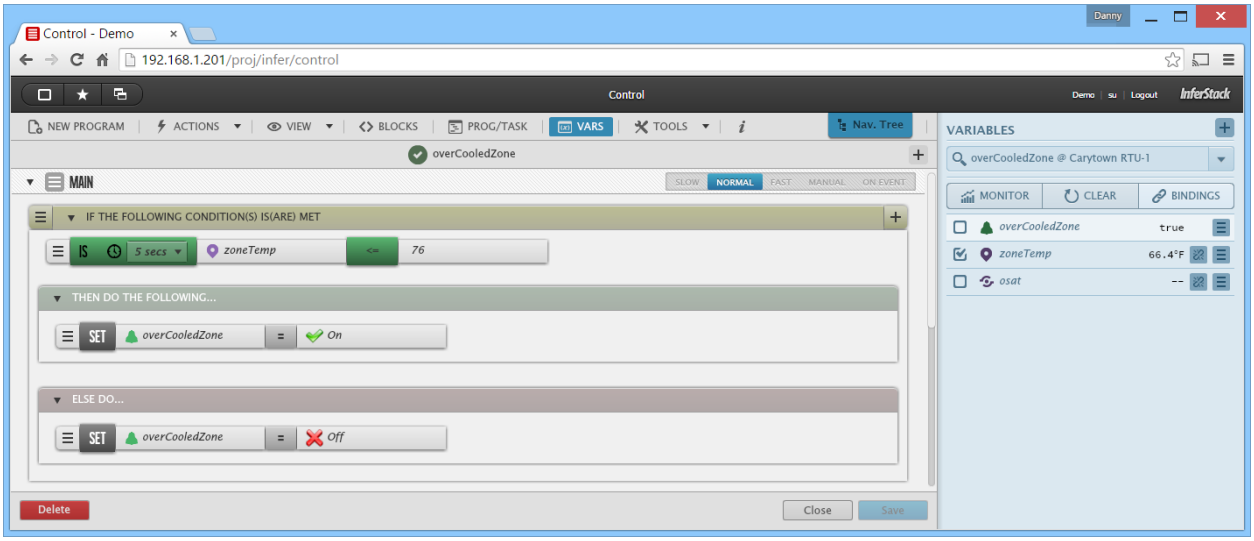


Bindings

Toggle variable bindings view.

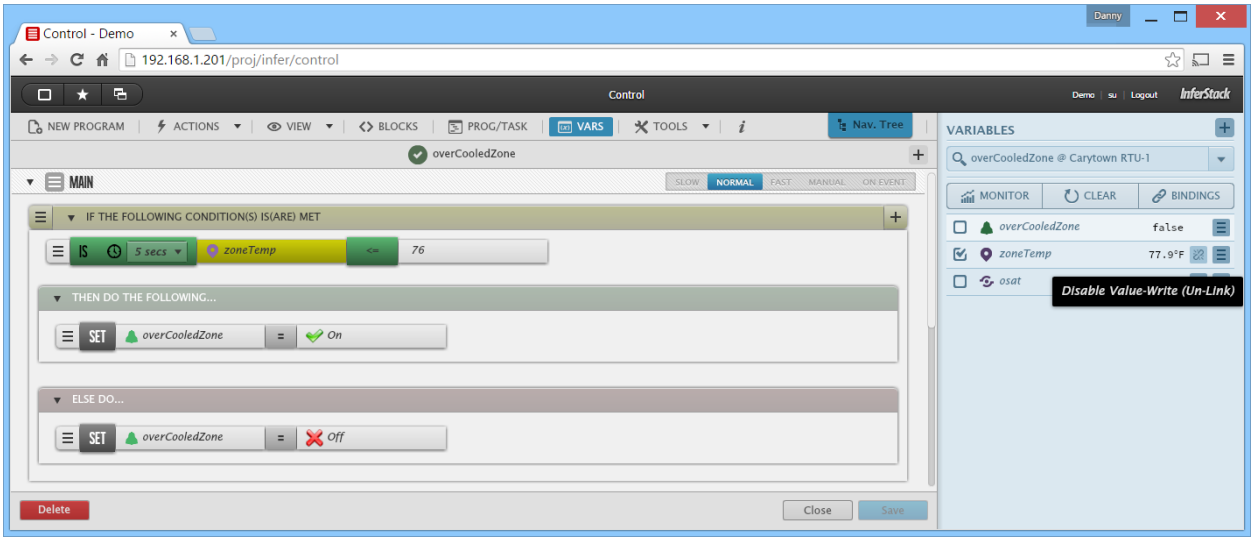






Disable Value Write (Unlink)

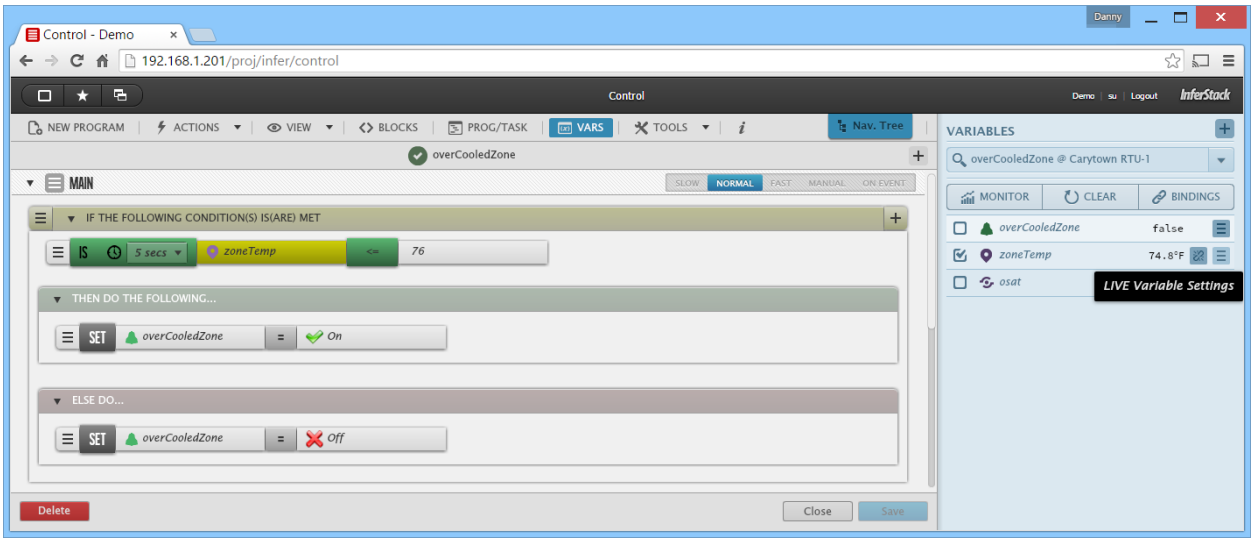
This allows you to decouple your program from the equipment the variables control.

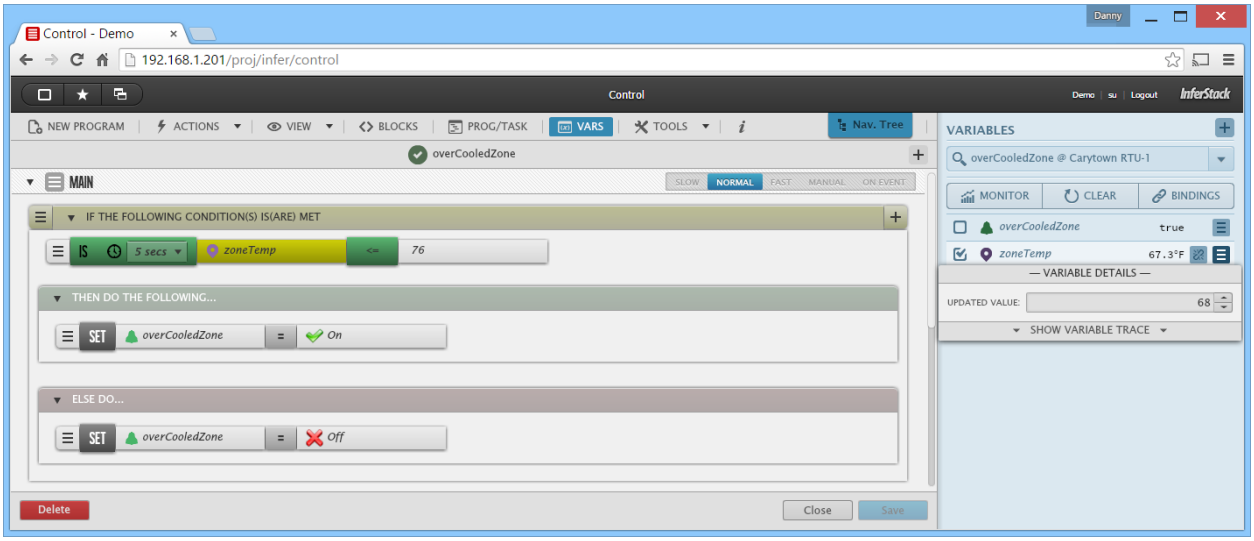


Live Variable settings

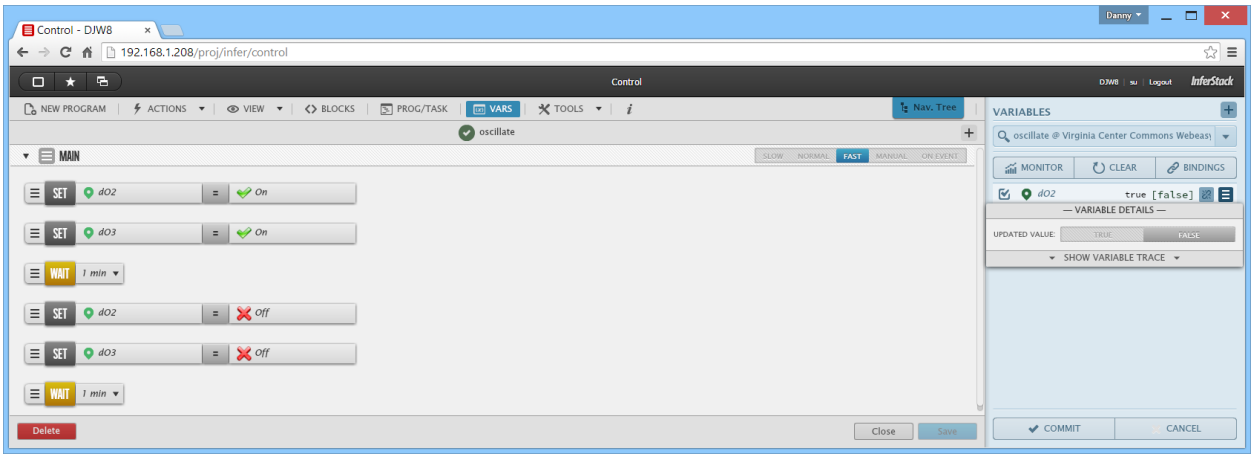
Press Live Variable settings to see variable details

.

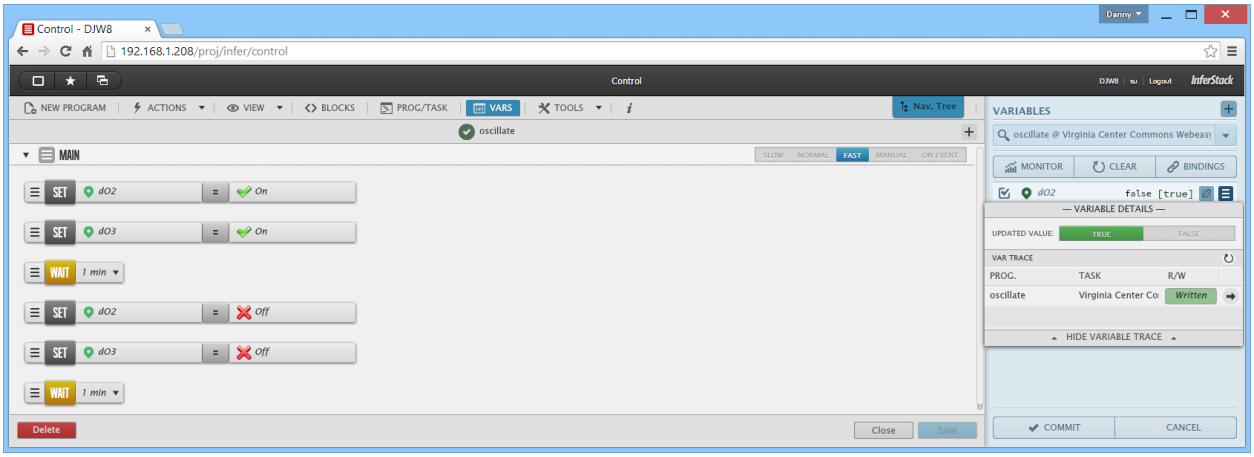




You can set values



And view variable trace.



Introduction to the Control Engine

Overview

The controlExt implements a control engine based upon the following principles:

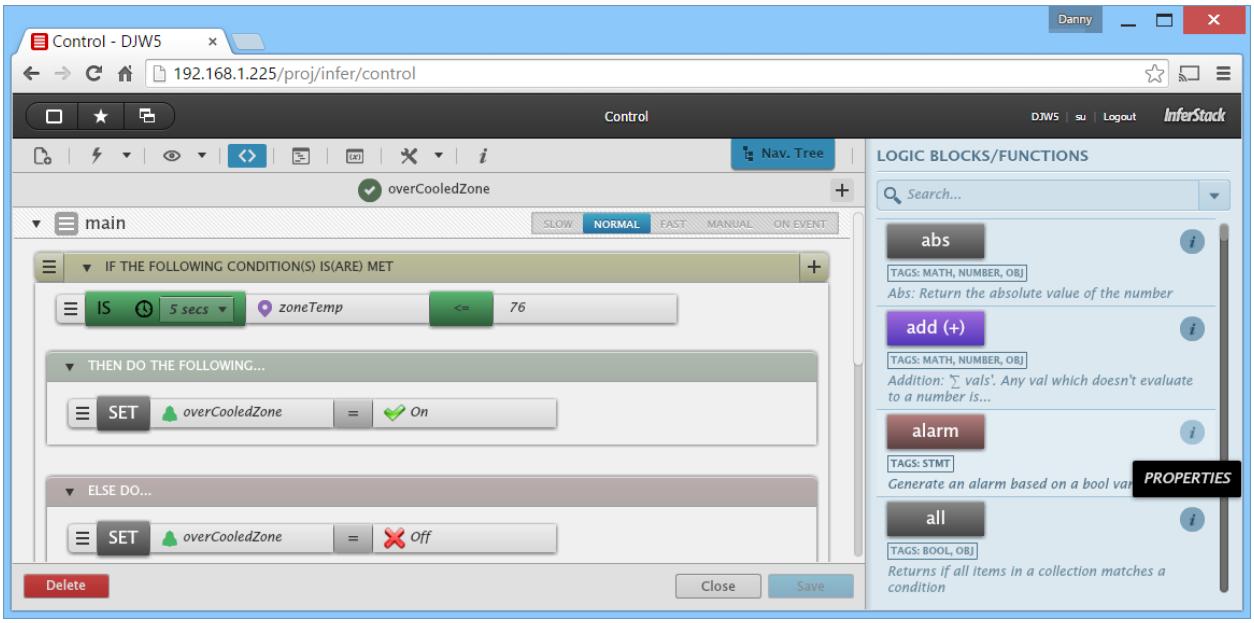
- *programs* implement control logic using a tree of *blocks*
- programs are organized into one or more *routines*
- routines are scheduled to run cyclically, on conditions, or manually
- programs are registered on targets using the `programOn` tag
- *tasks* are created for each unique program and target combination
- tasks maintain their state via *variables*
- local variables maintain state on a per task basis
- shared variables allow state to be shared between tasks
- bound variables allow state to be shared through folio and point writes
- bundle variables allow variable tables to be shared between programs

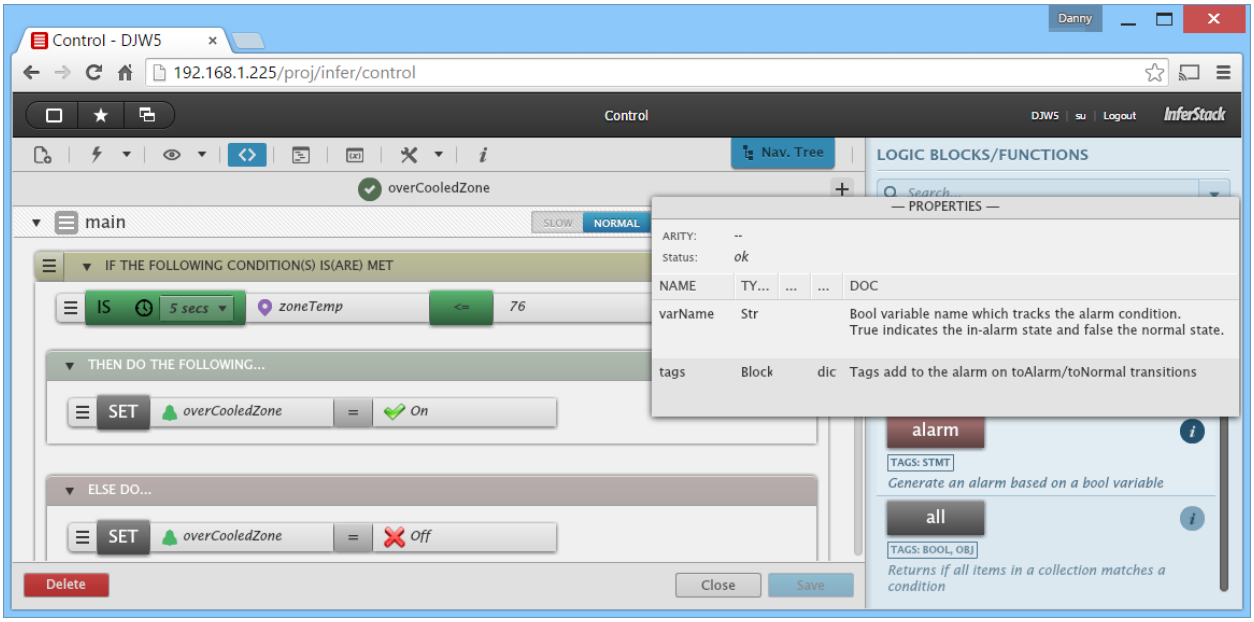
See http://licensing.intellastar.com/doc/ext-control/doc#overview for more information.

Library

You can press Blocks at any time you are viewing a program to see the available blocks.

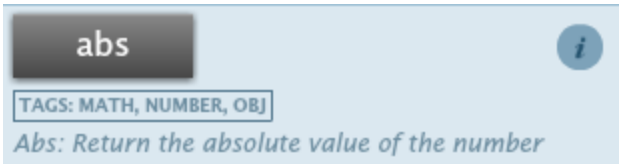
Press the Properties icon to view more details on the block.

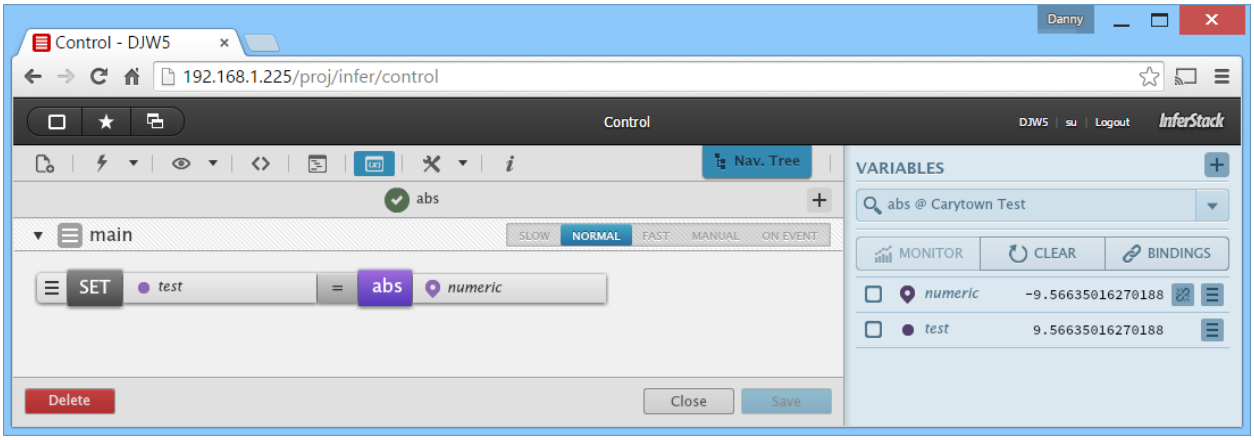




abs

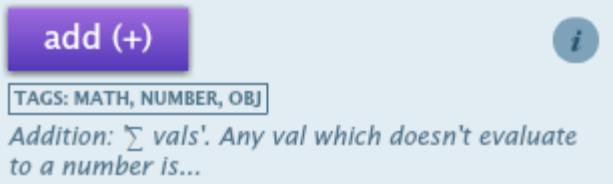
Abs: Return the absolute value of the number.





add

Addition: '∑ vals'. Any val which doesn't evaluate to a number is ignored. If no vals evaluate to a number then return null.

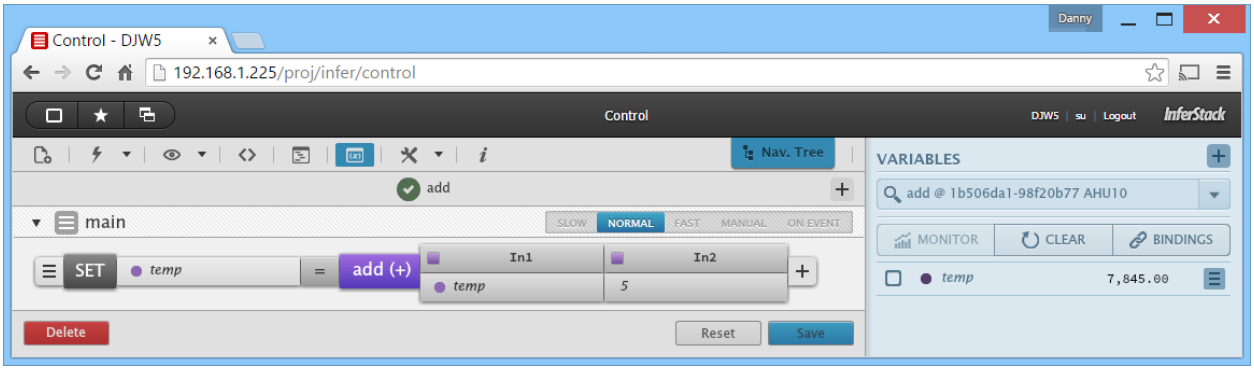


— PROPERTIES —

ARITY: *n*

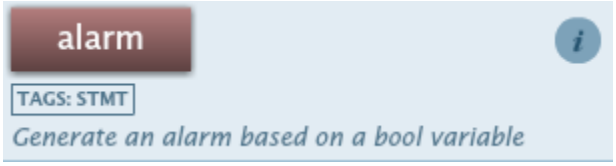
Status: *ok*

NAME	TYPE	Default	TAKES	DOC
vals	Block		number,gridCol,obj	



alarm

Generate an alarm based on a bool variable.



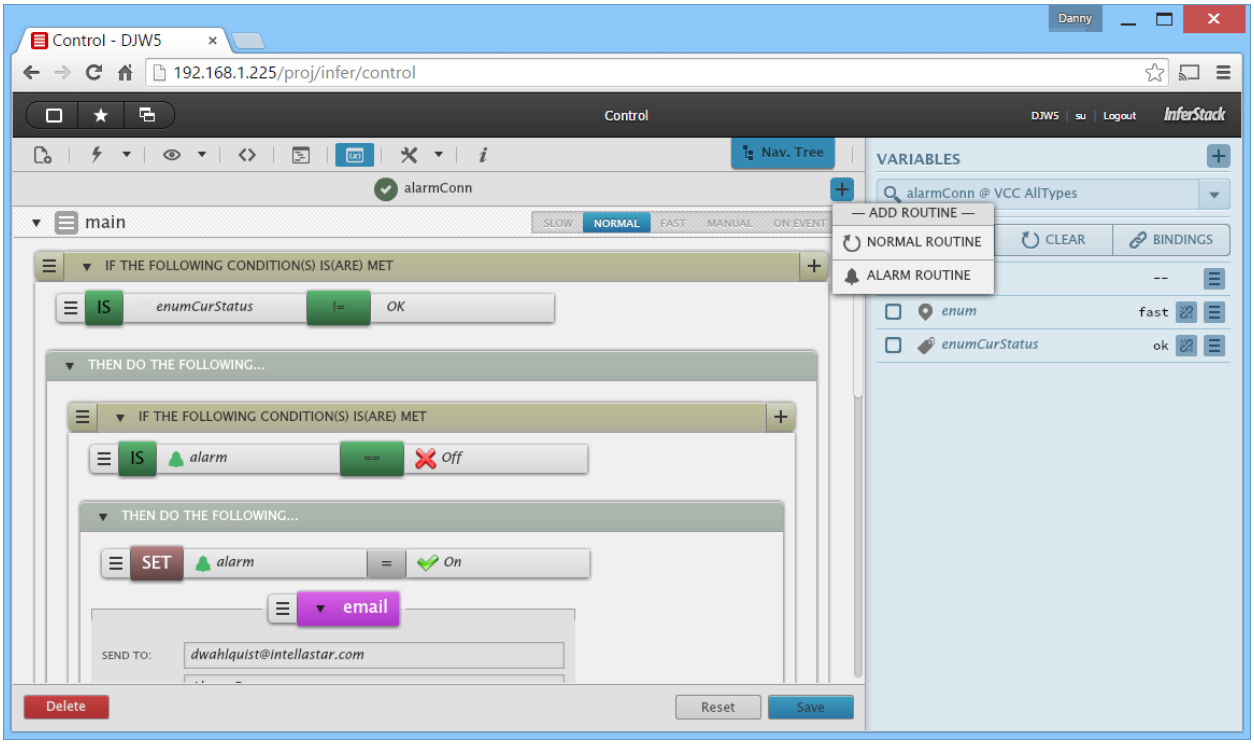
— PROPERTIES —

ARITY: --

Status: *ok*

NAME	TY...	...	...	DOC
varName	Str			Bool variable name which tracks the alarm condition. True indicates the in-alarm state and false the normal state.
tags	Block		dic	Tags add to the alarm on toAlarm/toNormal transitions

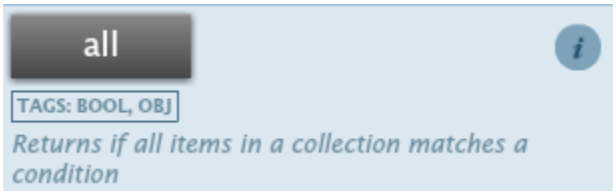
Press Add Routine and select Alarm Routine to use the Alarm Block.

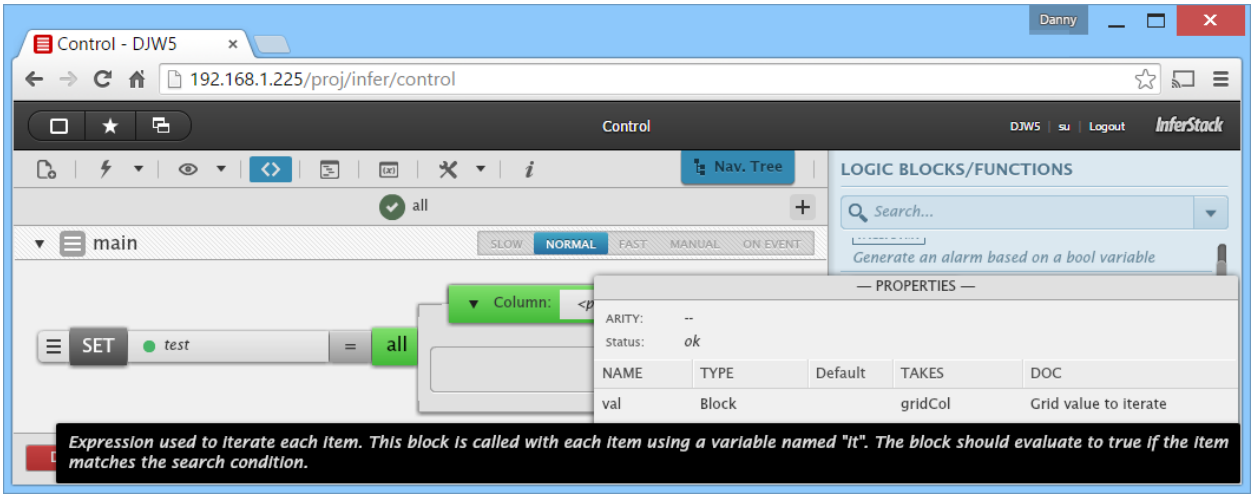


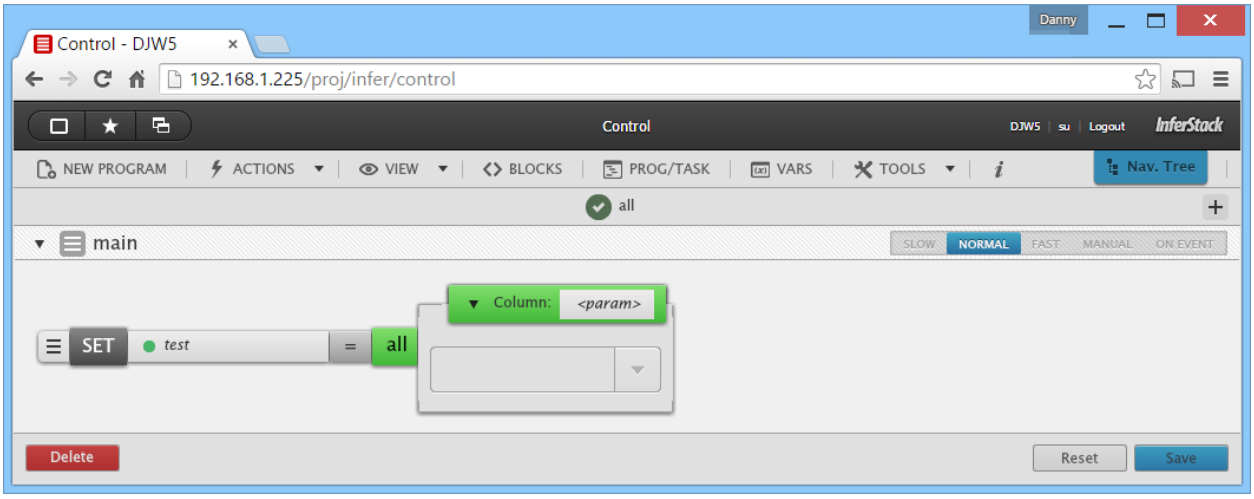
Note: The alarm email above is only sent when alarm is first set.

all

Returns if all items in a collection matches a condition

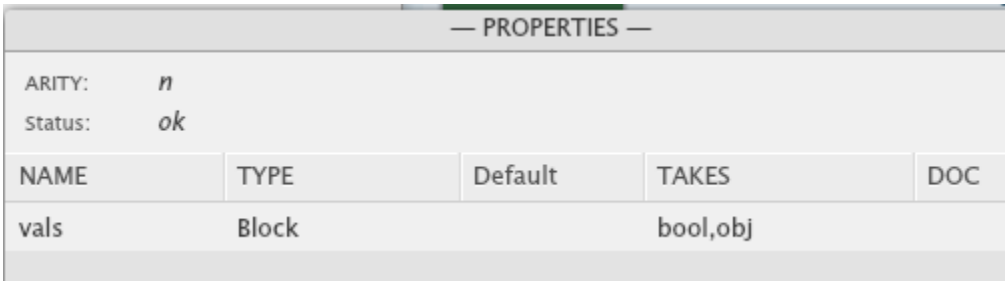


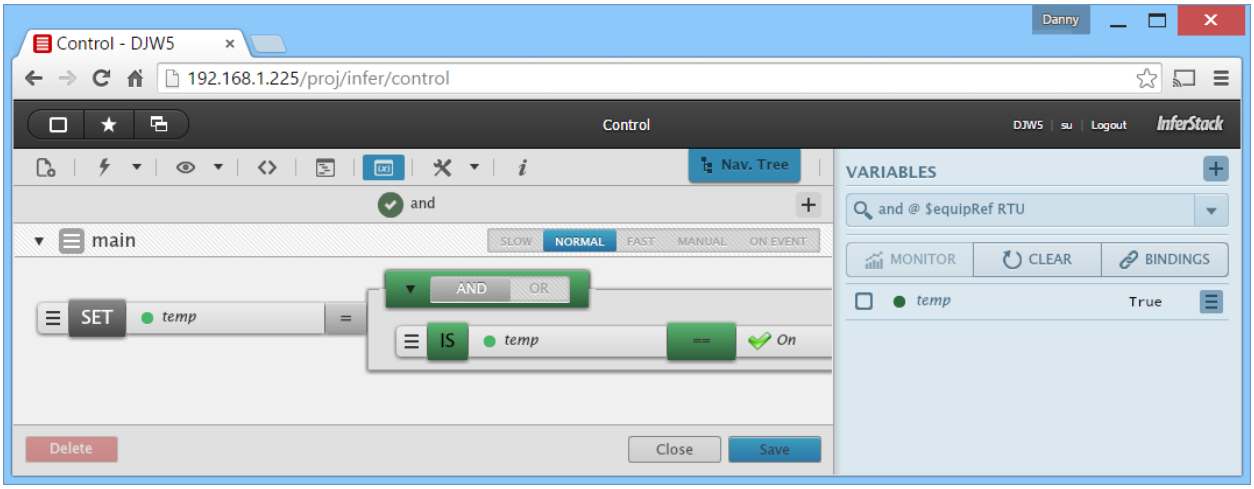




and

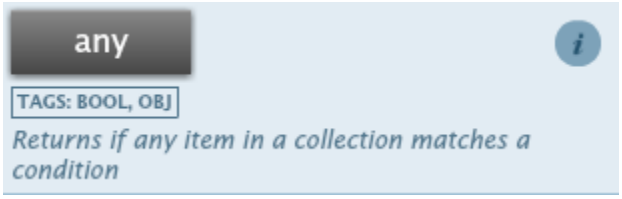
Logical-and of boolean values





any

Returns if any item in a collection matches a condition

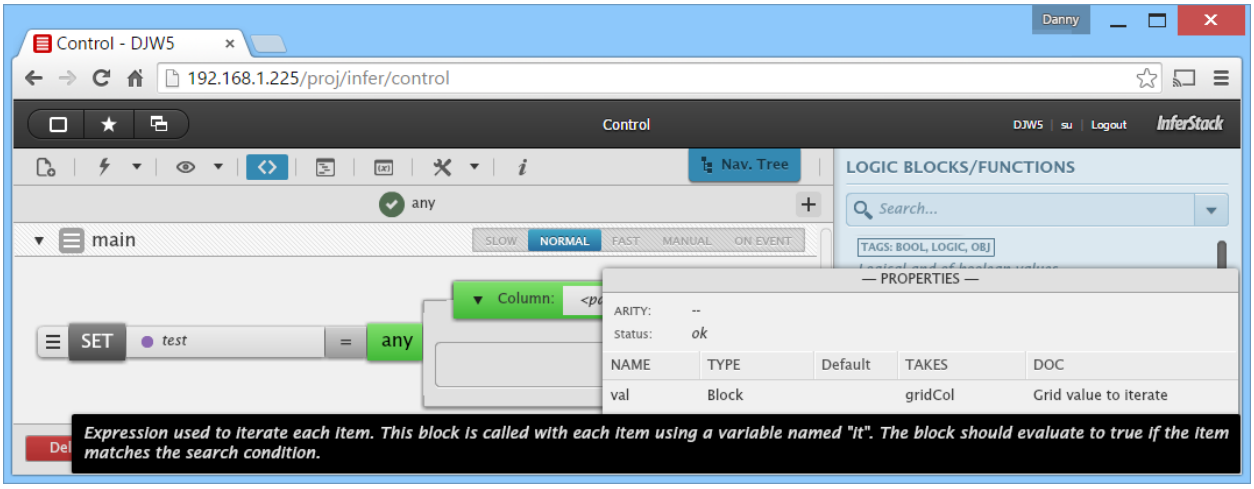


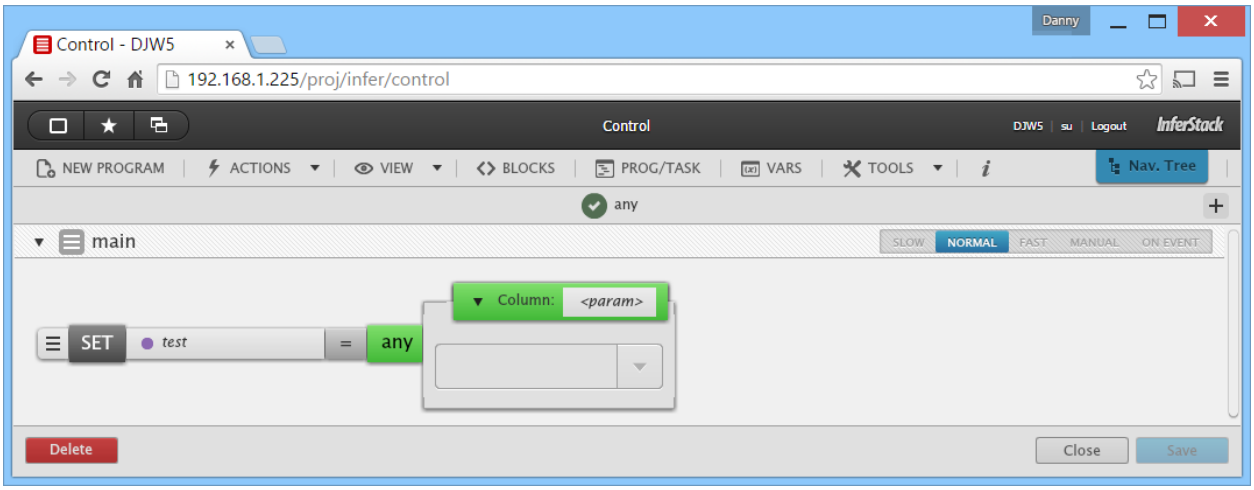
— PROPERTIES —

ARITY: --

Status: *ok*

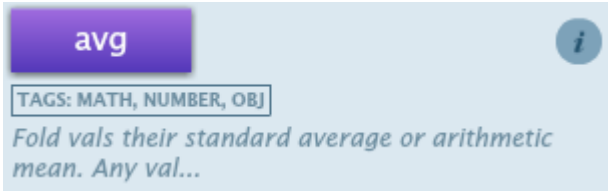
NAME	TYPE	Default	TAKES	DOC
val	Block		gridCol	Grid value to iterate
iterator	Block		bool,obj	Expression used to iterate ea item using a variable named '





avg

Fold vals their standard average or arithmetic mean. Any val which doesn't evaluate to a number is ignored. If no vals evaluate to a number then return null.

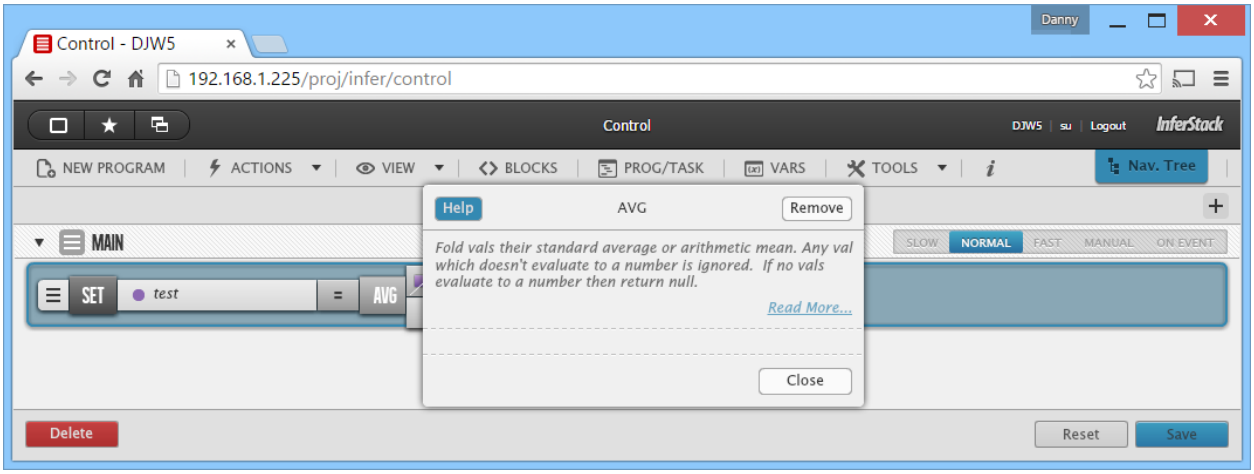


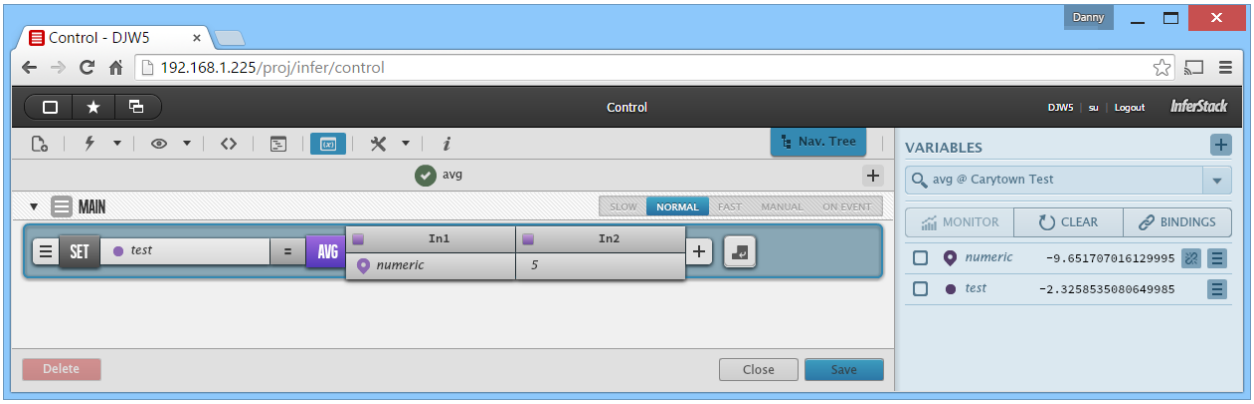
— PROPERTIES —

ARITY: *n*

Status: *ok*

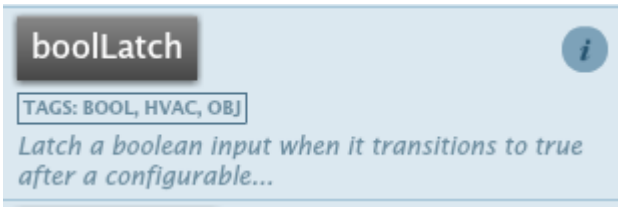
NAME	TYPE	Default	TAKES	DOC
vals	Block		number,gridCol,obj	





boolLatch

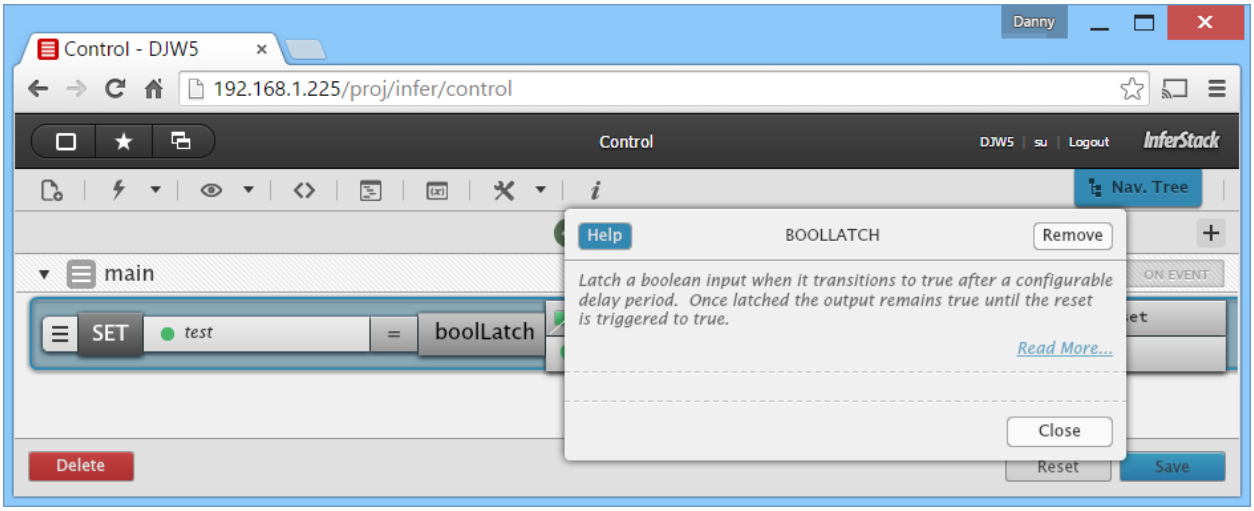
Latch a Boolean input when it transitions to true after a configurable delay period. Once latched the output remains true until the reset is triggered to true.

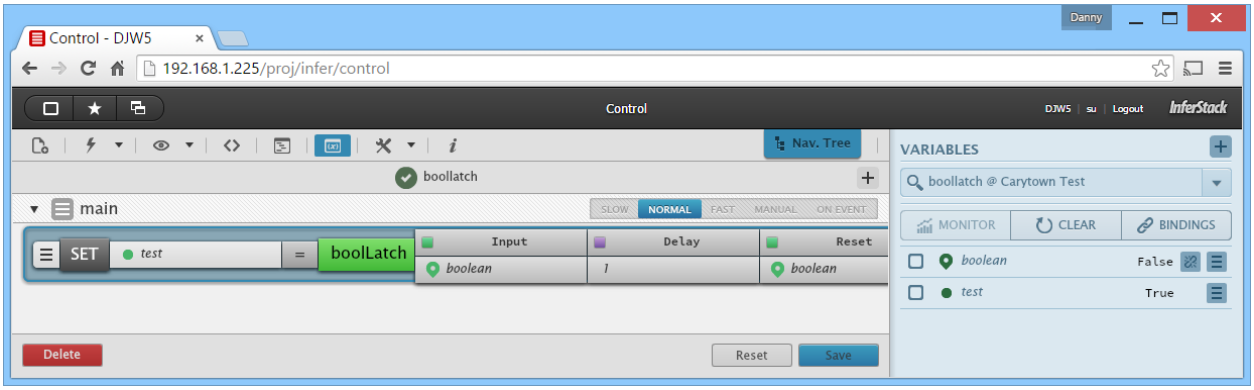


— PROPERTIES —

ARITY: --
Status: *ok*

NAME	TYPE	Default	TAKES	DOC
input	Block		bool,obj	Input condition to monitor
delay	Block		number,obj	Duration number for how long inp
reset	Block		bool,obj	Reset output back to false

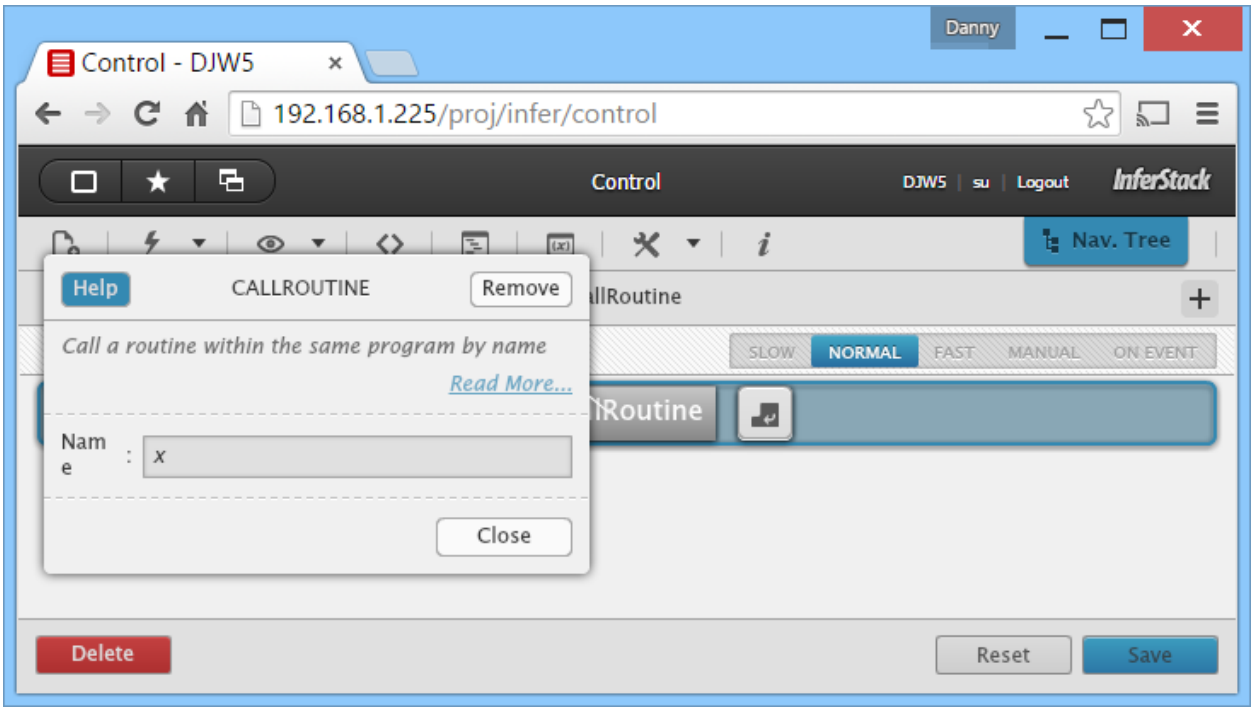




callRoutine

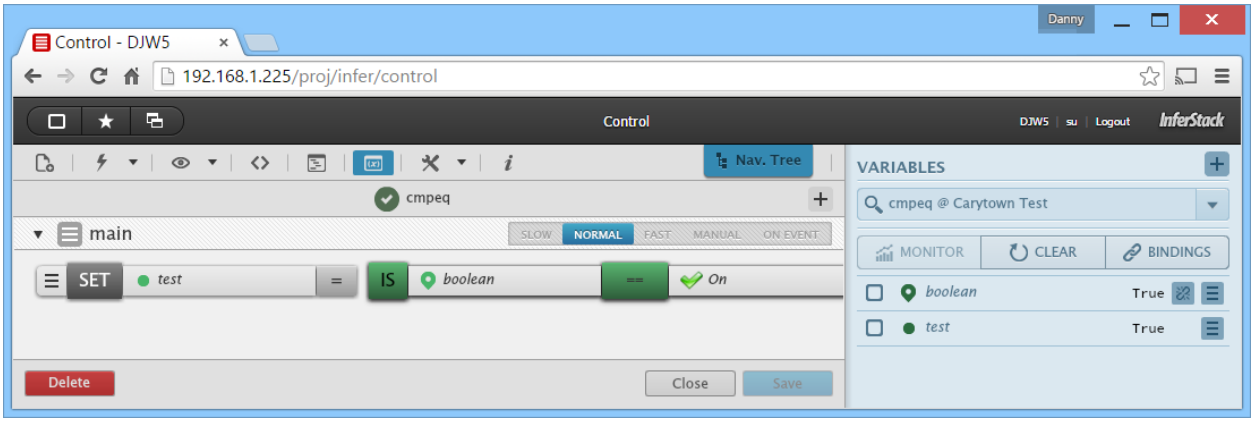
Call a routine within the same program by name

Note: When you implement routines within a program that act like subroutines using "callRoutine", the routine run type must be flagged as "manual". Otherwise the routines run all the time.



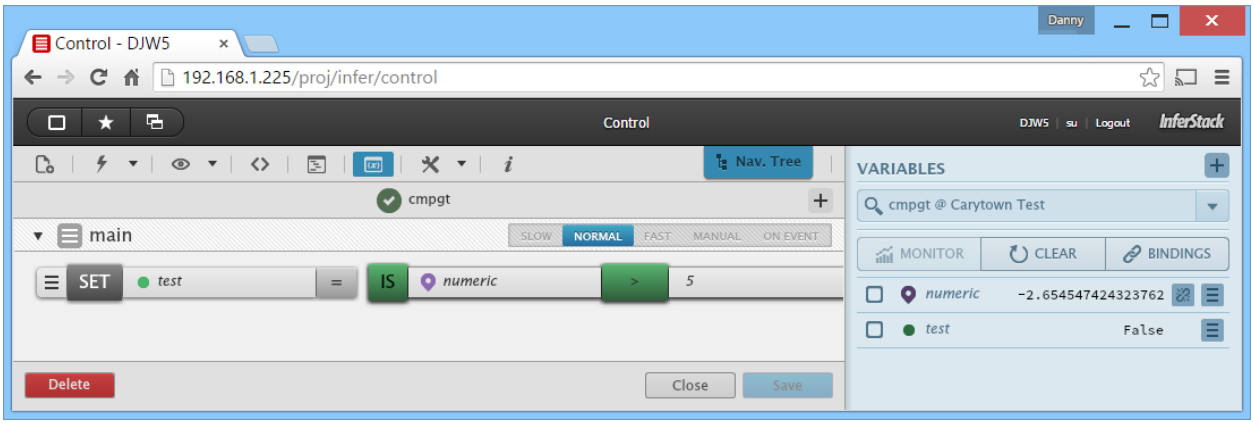
cmpEq

Equality comparison



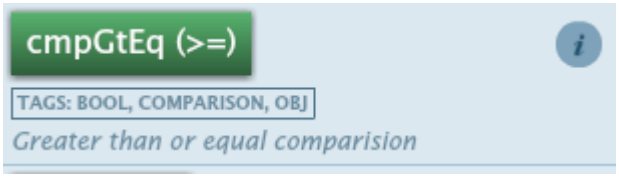
cmpGt

Greater than comparision



cmpGtEq

Greater than or equal comparision



cmpLt

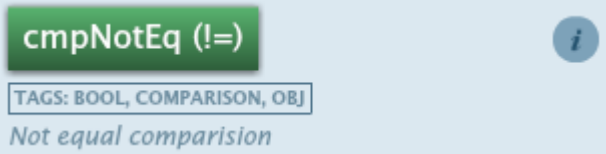
Less than comparision

cmpLtEq

Less than or equal comparision

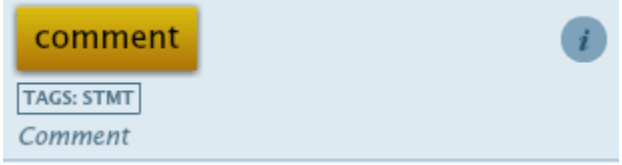
cmpNotEq

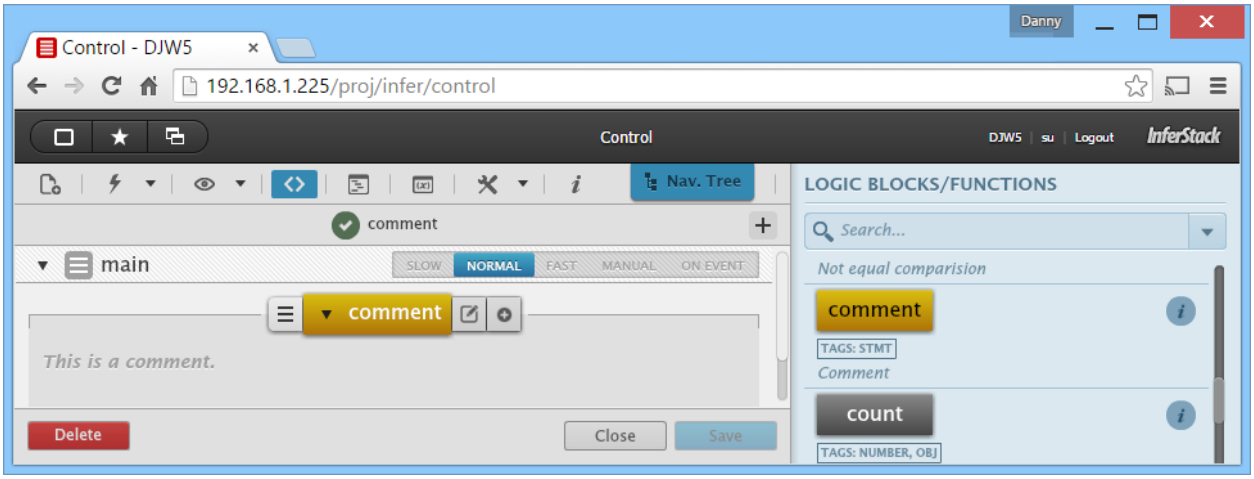
Not equal comparision



comment

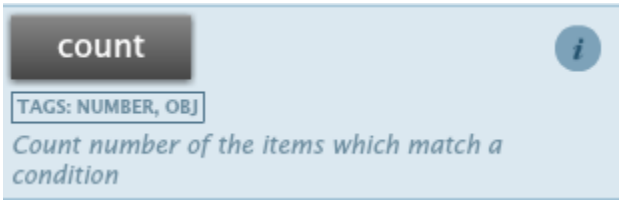
Comment

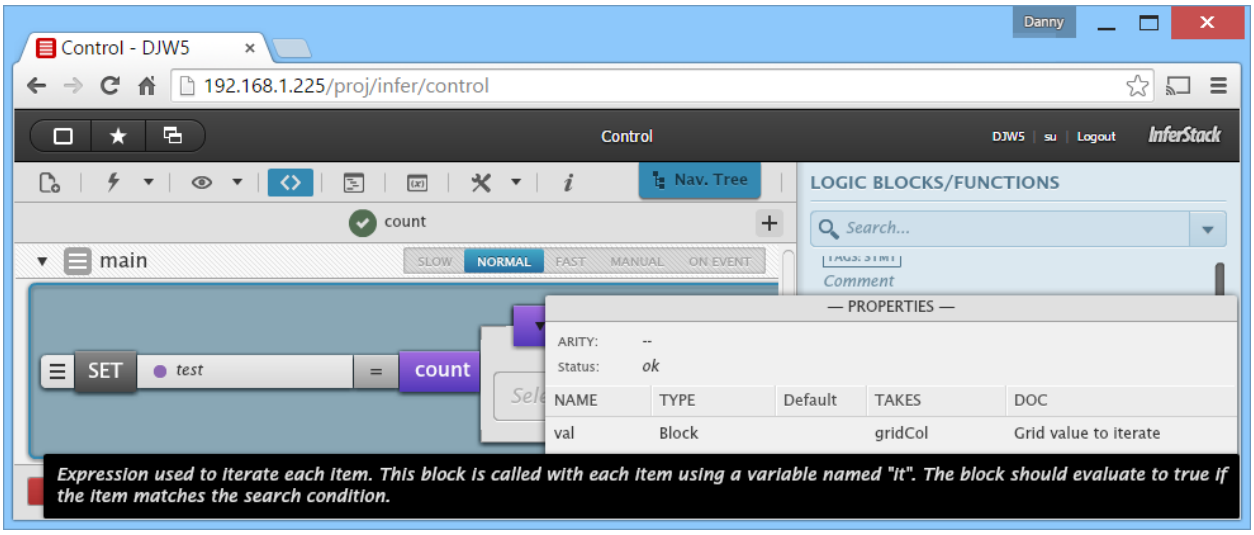


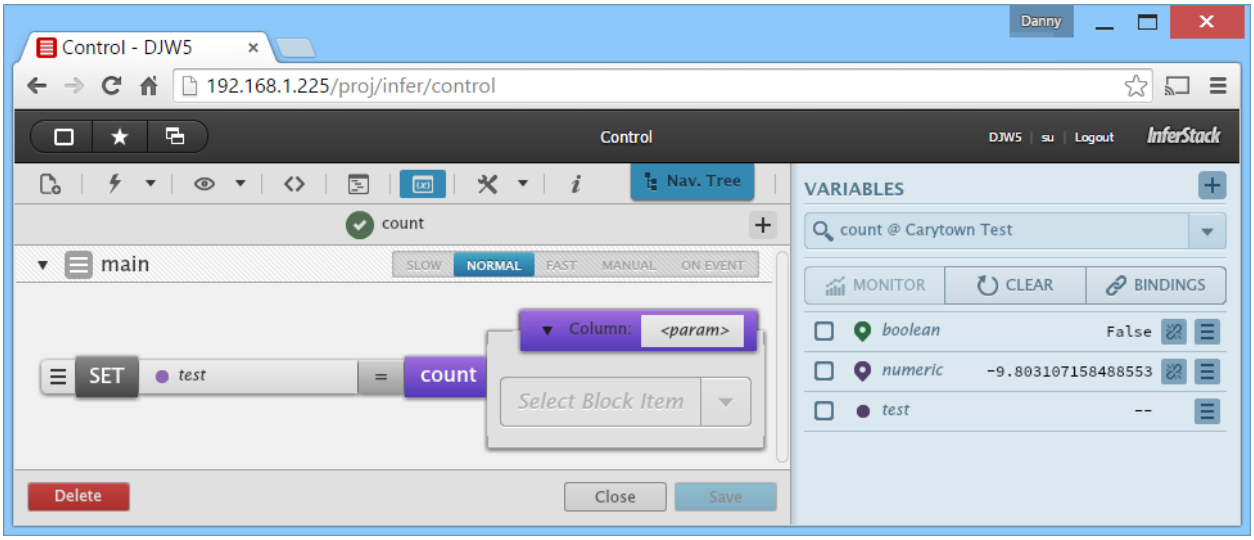


count

Count number of the items which match a condition

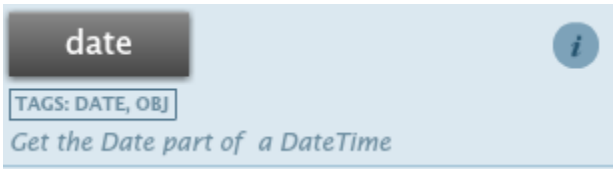


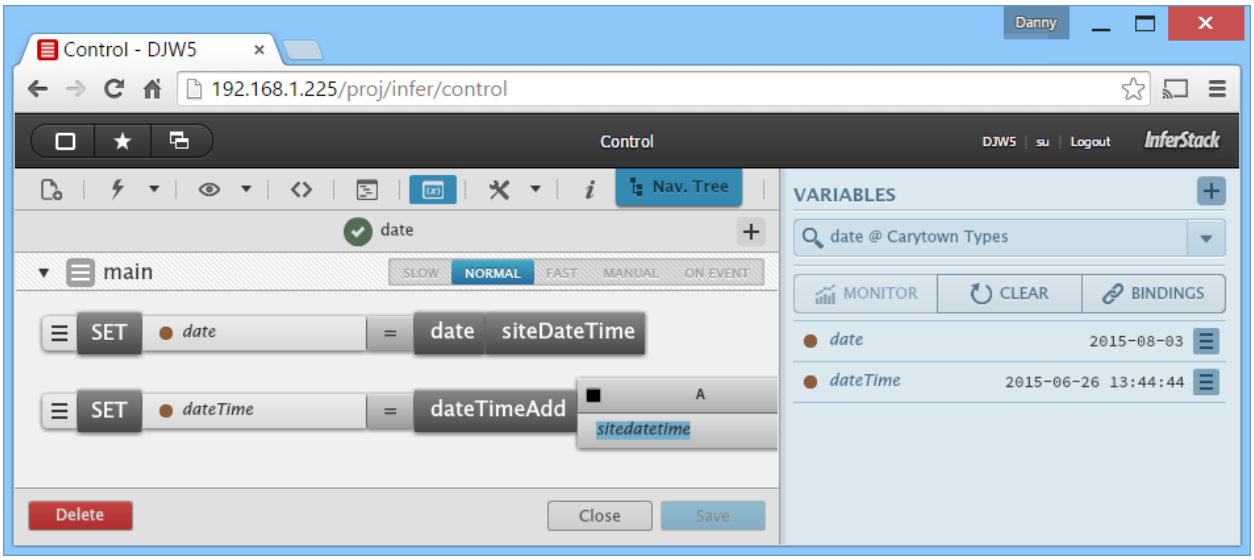




date

Get the Date part of a DateTime.





dateTimeAdd

Add a duration from a DataTime. If no units are supplied, then minutes will be assumed.

dateTimeAdd

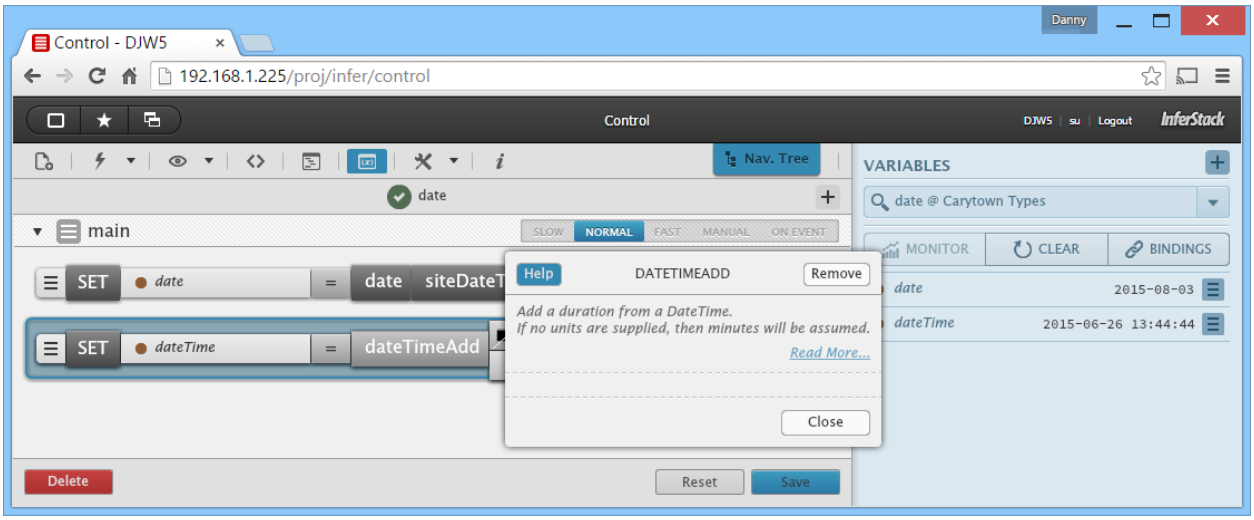
TAGS: DATETIME, OBJ

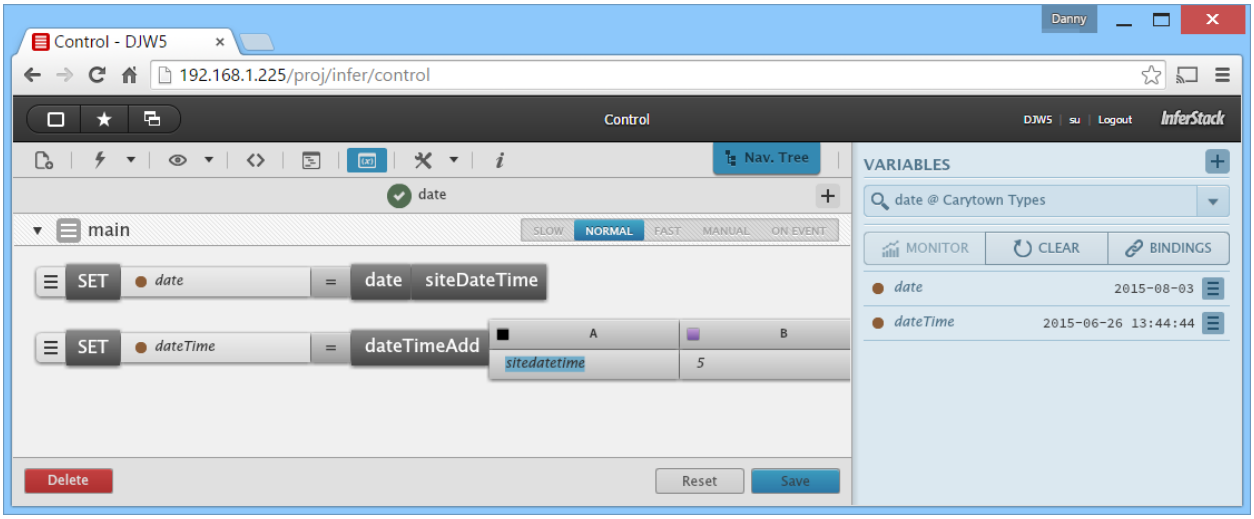
Add a duration from a DateTime.
If no units are supplied, then minutes will be ass...

— PROPERTIES —

ARITY: *binary*
Status: *ok*

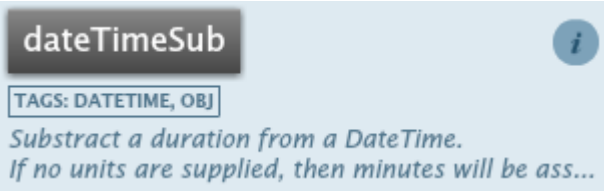
NAME	TYPE	Default	TAKES
a	Block		dateTime,obj
b	Block		number,obj





dateTimeSub

Subtract a duration from a DataTime. If no units are supplied, then minutes will be assumed.

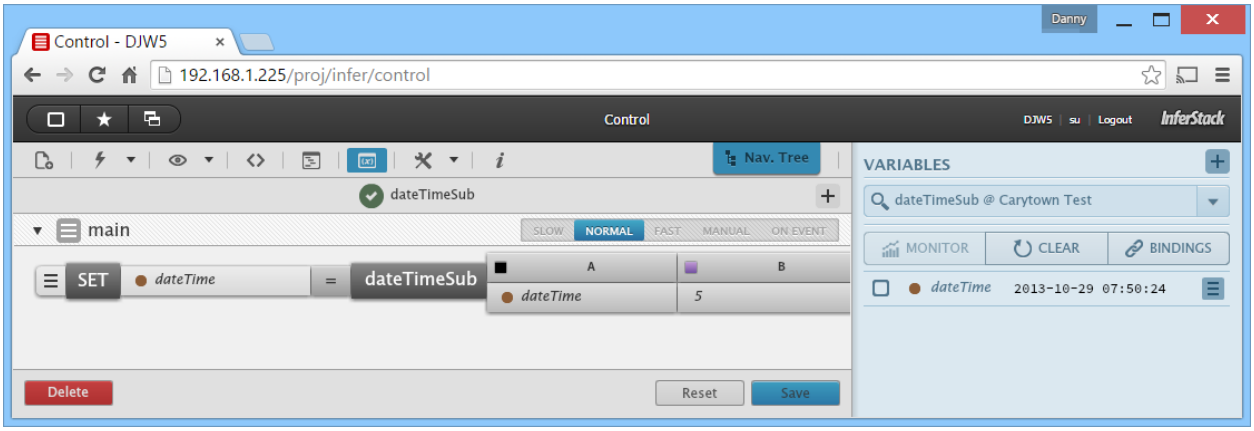


— PROPERTIES —

ARITY: *binary*

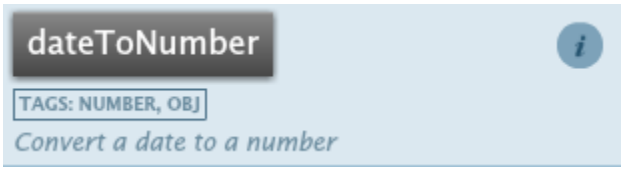
Status: *ok*

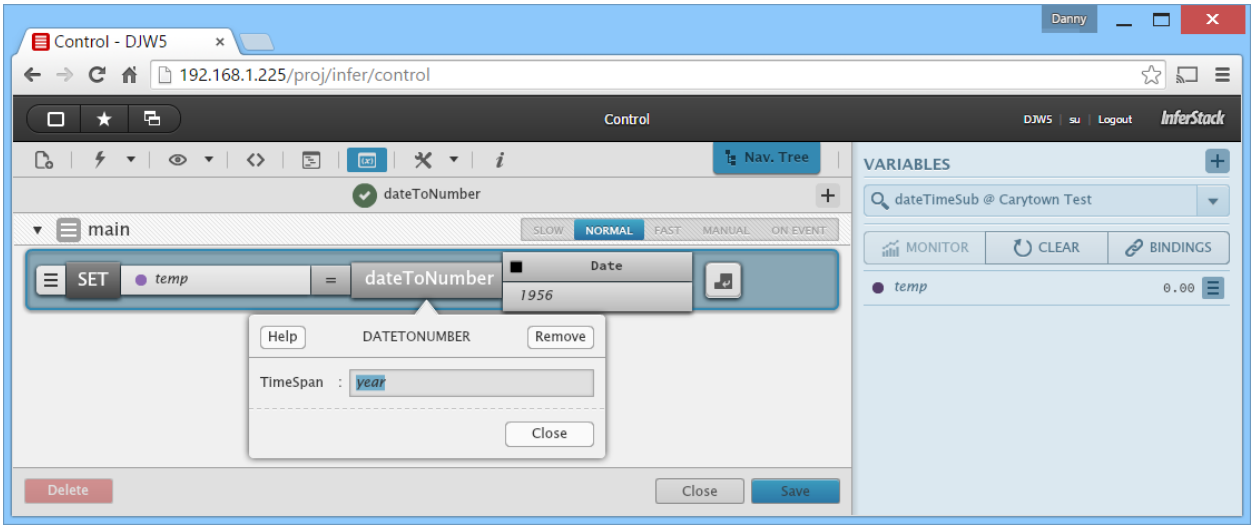
NAME	TYPE	Default	TAKES
a	Block		dateTime,obj
b	Block		number,obj



dateToNumber

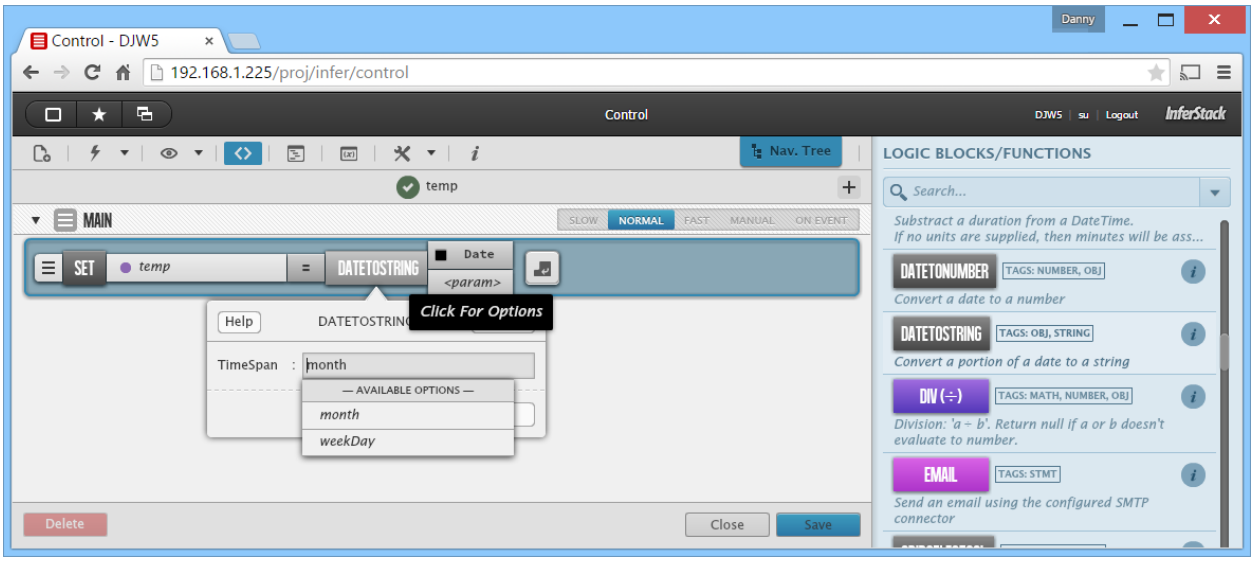
Convert a date to a number.





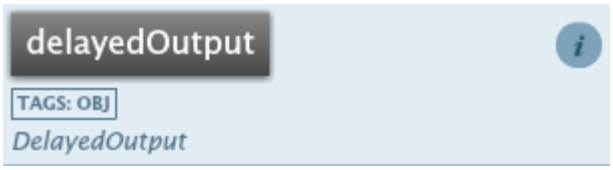
dateToString

Convert a portion of a date to a string.



delayedOutput

Delayed Output

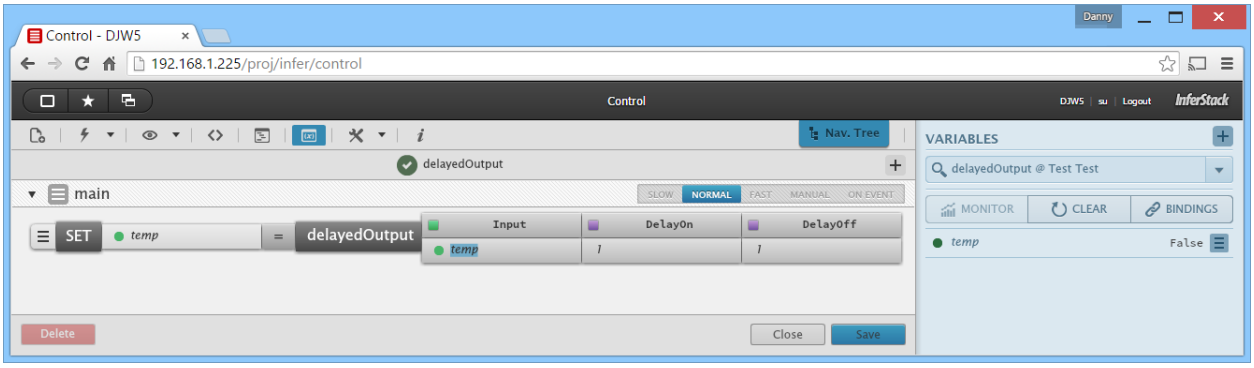


— PROPERTIES —

ARITY: --

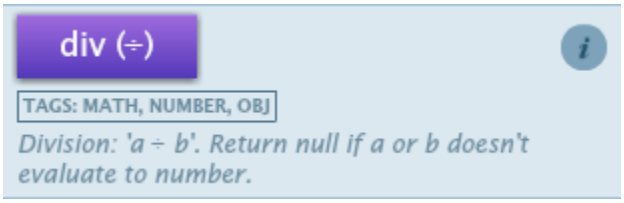
Status: *ok*

NAME	TYPE	Default	TAKES	DOC
input	Block		bool,obj	Input condition to monitor
delayOn	Block		number,obj	Duration number for how long inp
delayOff	Block		number,obj	Duration number for how long inp



div

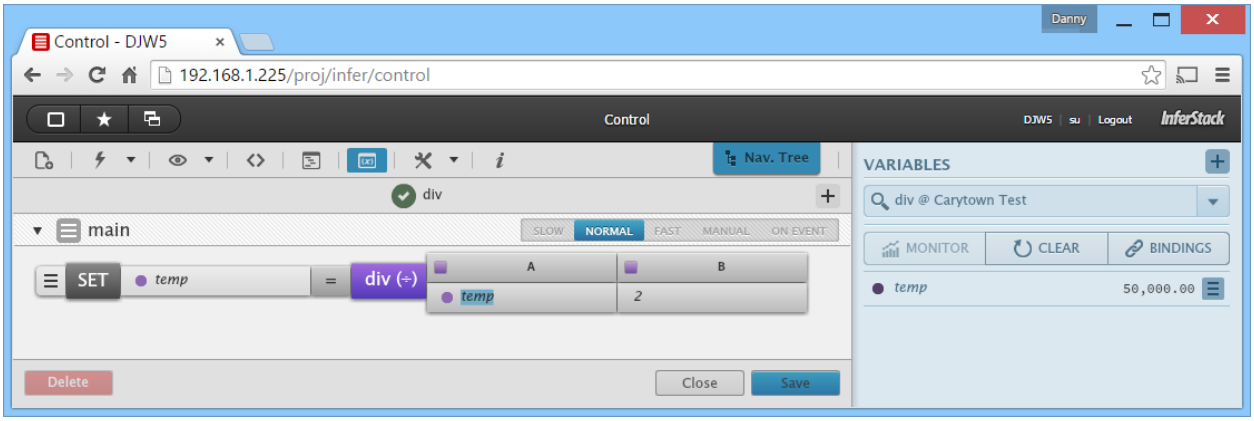
Division: 'a ÷ b'. Return null if a or b doesn't evaluate to number.



— PROPERTIES —

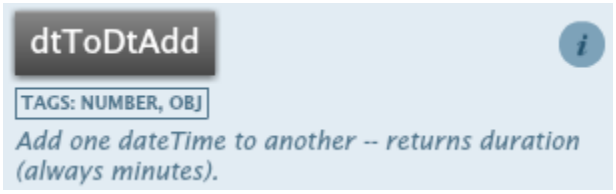
ARITY: *binary*
Status: *ok*

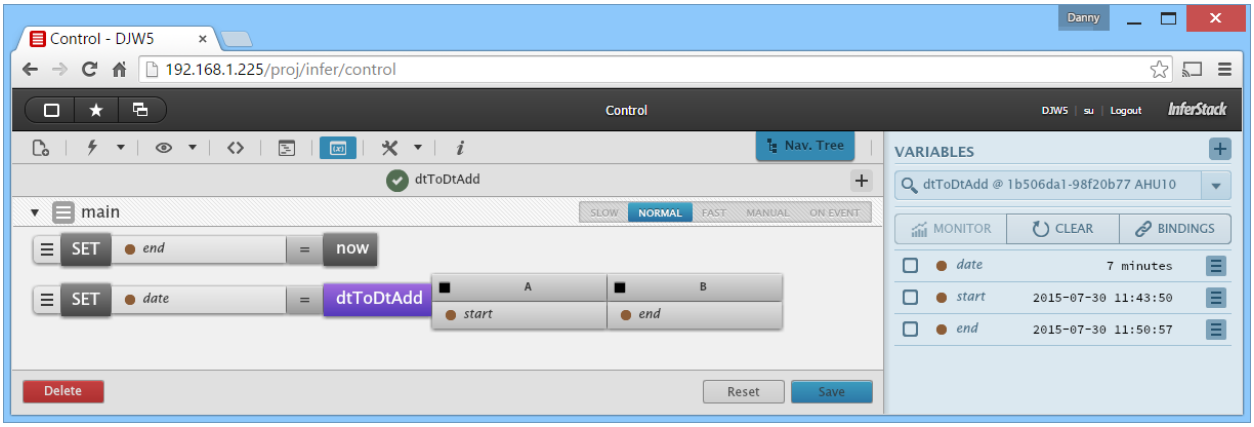
NAME	TYPE	Default	TAKES	DOC
a	Block		number,obj	
b	Block		number,obj	



dtToDtAdd

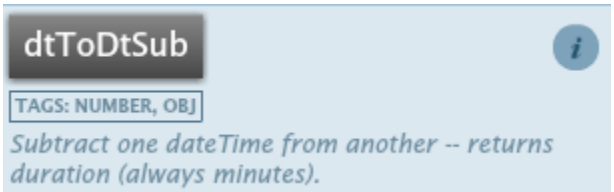
Add one dateTime to another – returns duration (always minutes).

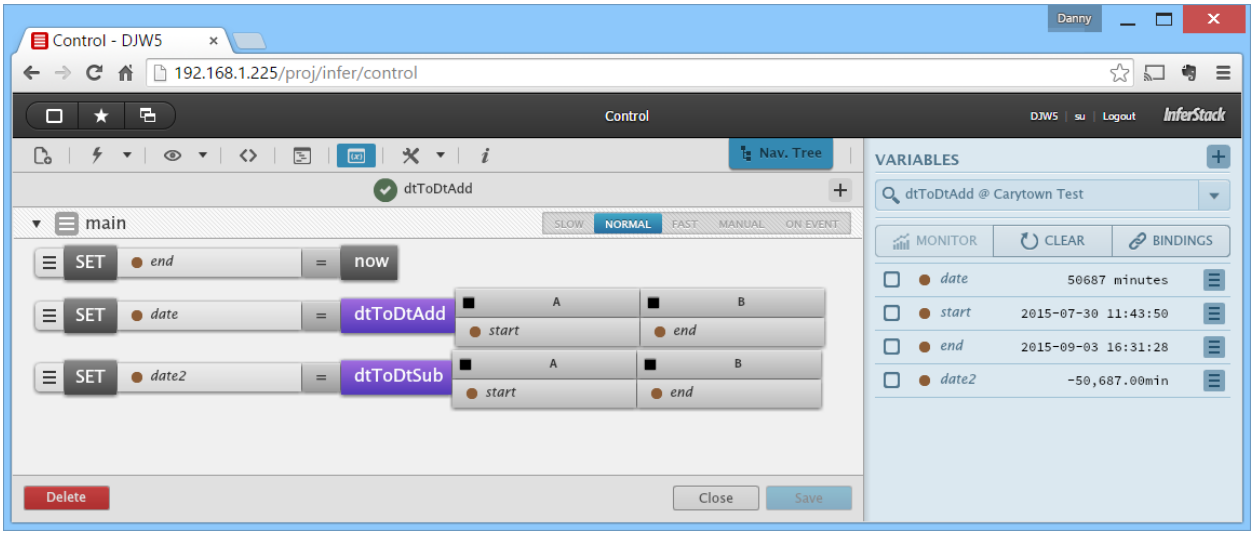




dtToDtSub

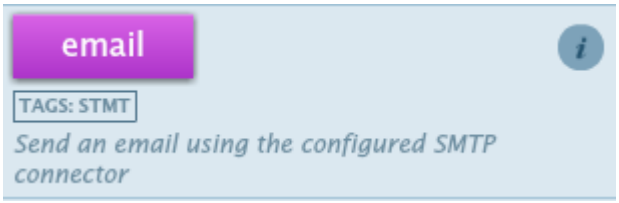
Subtract one dateTime from another – returns duration (always minutes).





email

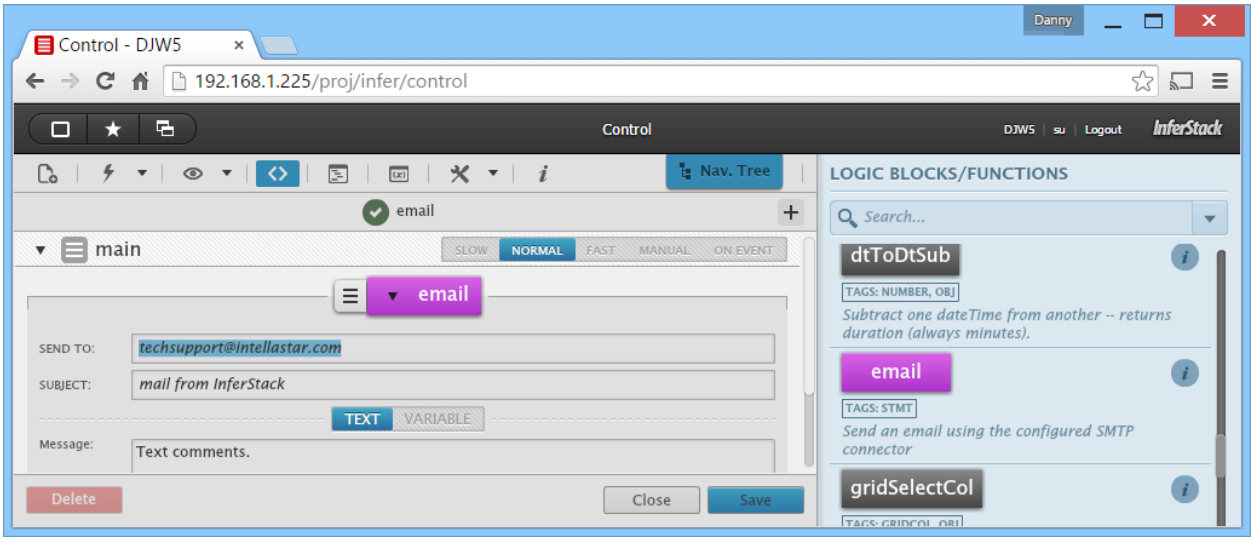
Send an email using the configured SMTP connector



— PROPERTIES —

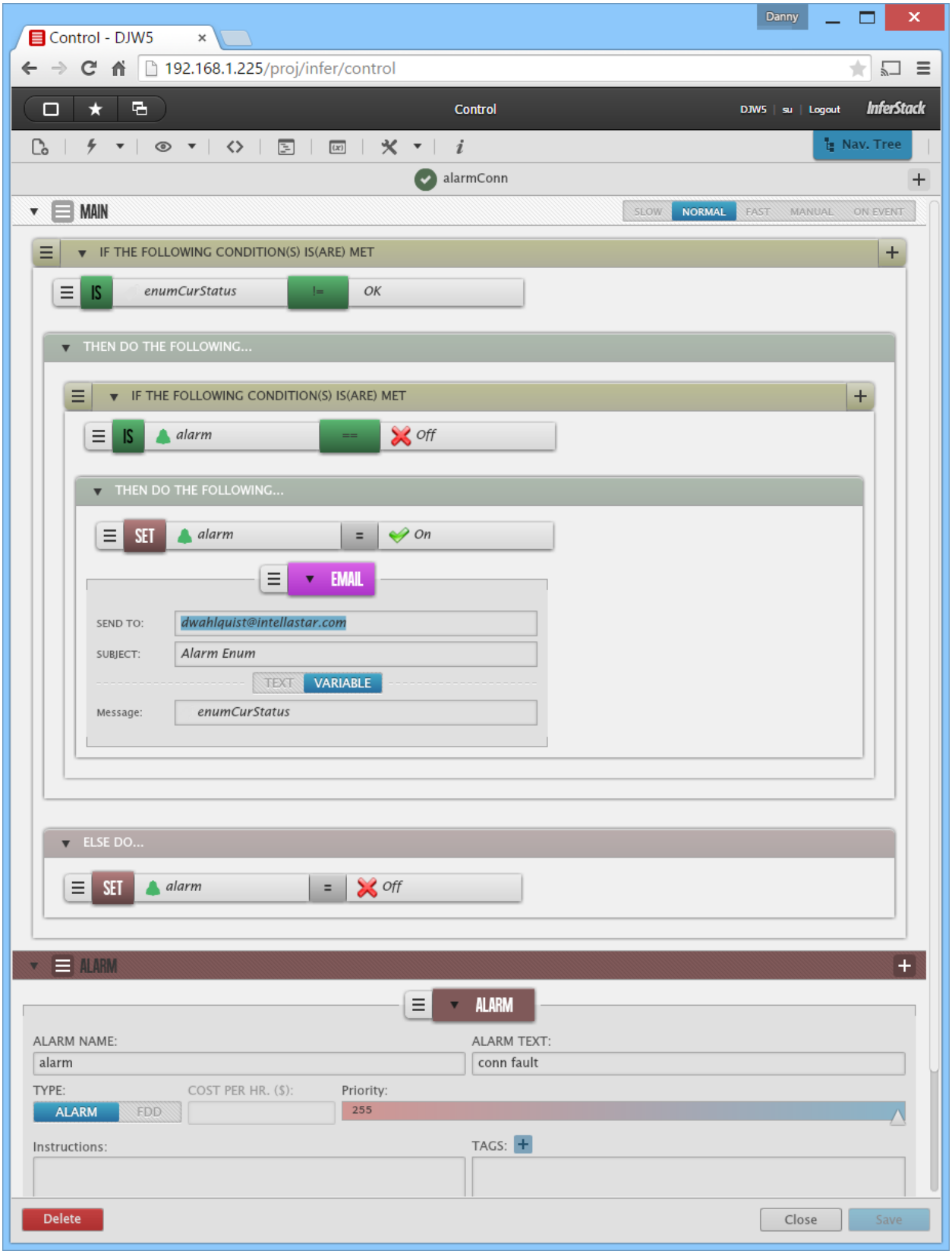
ARITY: --
Status: ok

NAME	TYPE	D...	TAKES	DOC
to	Block		str,obj	List of 'to' email addresses separated by comma
subjec	Block		str,obj	Subject line
body	Block		str,obj	Email content as either plain text or HTML string



Email Alarm

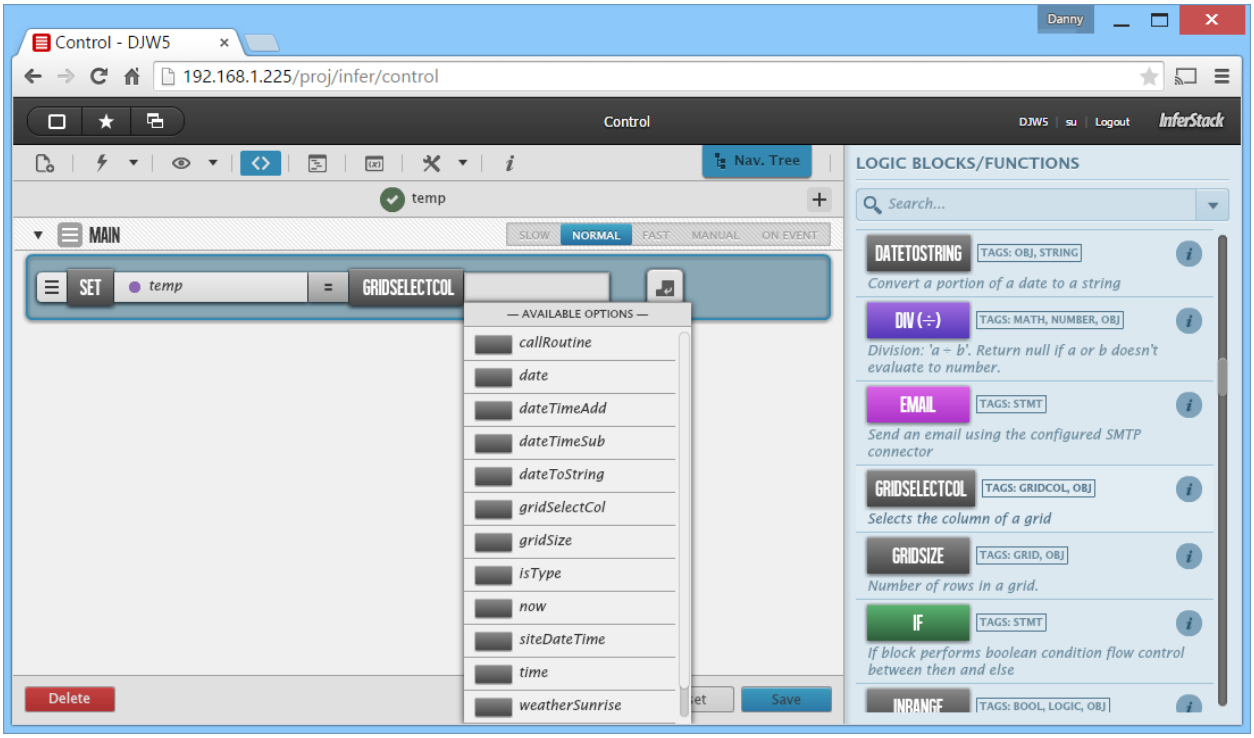
When curStatus is not ok (and we are not already in alarm), we set the alarm and add an email block to send an email. You can send text, or as shown here, you can select a variable (curStatus)


Danny
Control - DJW5
192.168.1.225/proj/infer/control
Control
DJW5 | su | Logout
InferStack
Nav. Tree
alarmConn
MAIN
SLOW NORMAL FAST MANUAL ON EVENT
IF THE FOLLOWING CONDITION(S) IS(ARE) MET
IS enumCurStatus != OK
THEN DO THE FOLLOWING...
IF THE FOLLOWING CONDITION(S) IS(ARE) MET
IS alarm == Off
THEN DO THE FOLLOWING...
SET alarm = On
EMAIL
SEND TO: dwahlquist@intellastar.com
SUBJECT: Alarm Enum
TEXT VARIABLE
Message: enumCurStatus
ELSE DO...
SET alarm = Off
ALARM
ALARM
ALARM NAME: alarm
ALARM TEXT: conn fault
TYPE: ALARM FDD
COST PER HR. ($):
Priority: 255
Instructions:
TAGS:
Delete
Close
Save

Note: The alarm email above is only sent when alarm is first set.

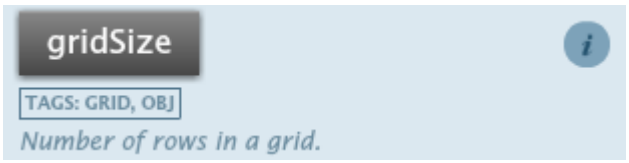
gridSelectCol

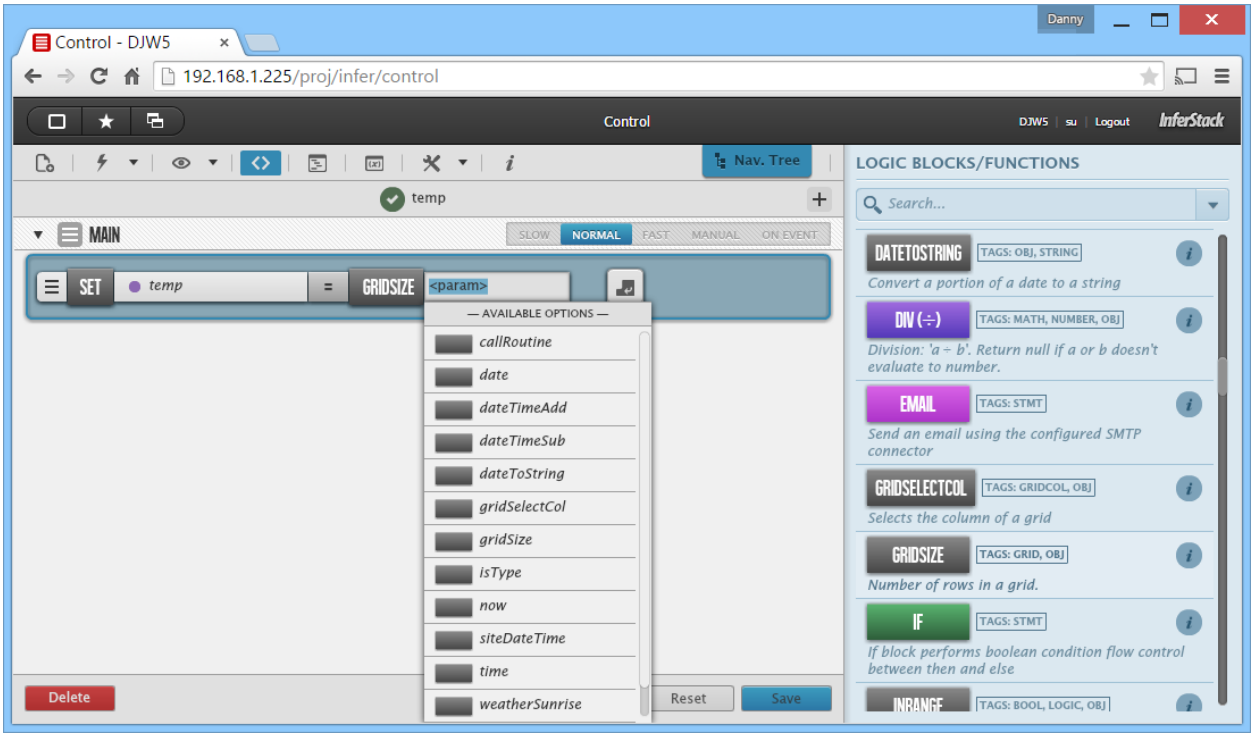
Selects the column of a grid



gridSize

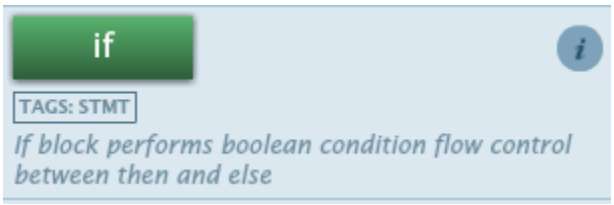
Number of rows in a grid.





if

If block performs boolean condition flow control between then and else



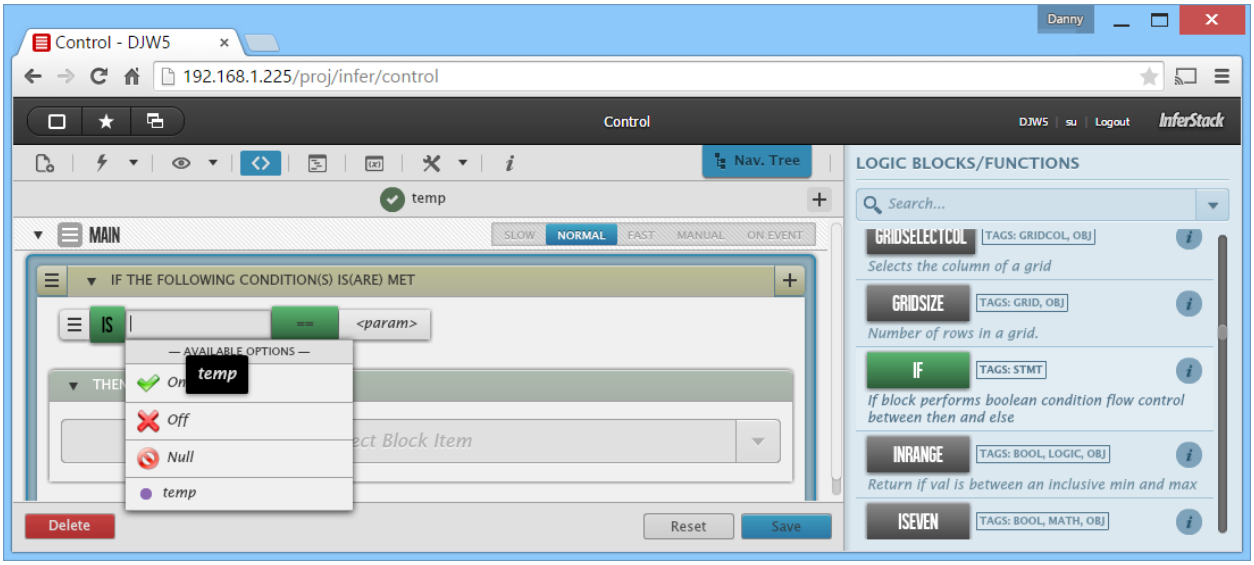
— PROPERTIES —

ARITY: --

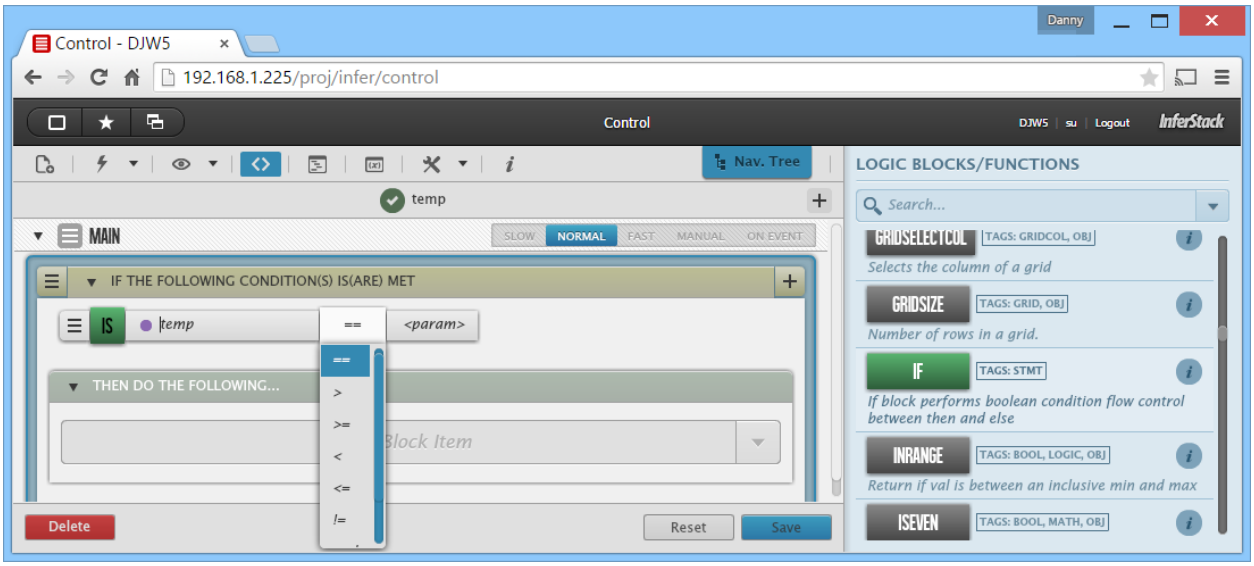
Status: *ok*

NAME	TYPE	Def...	TAKES	DOC
cond	Block		bool,obj	Condition used to determine branch
then	Block		stmt	Statement to run when cond is true
else	Block		stmt	Statement to run when cond is false

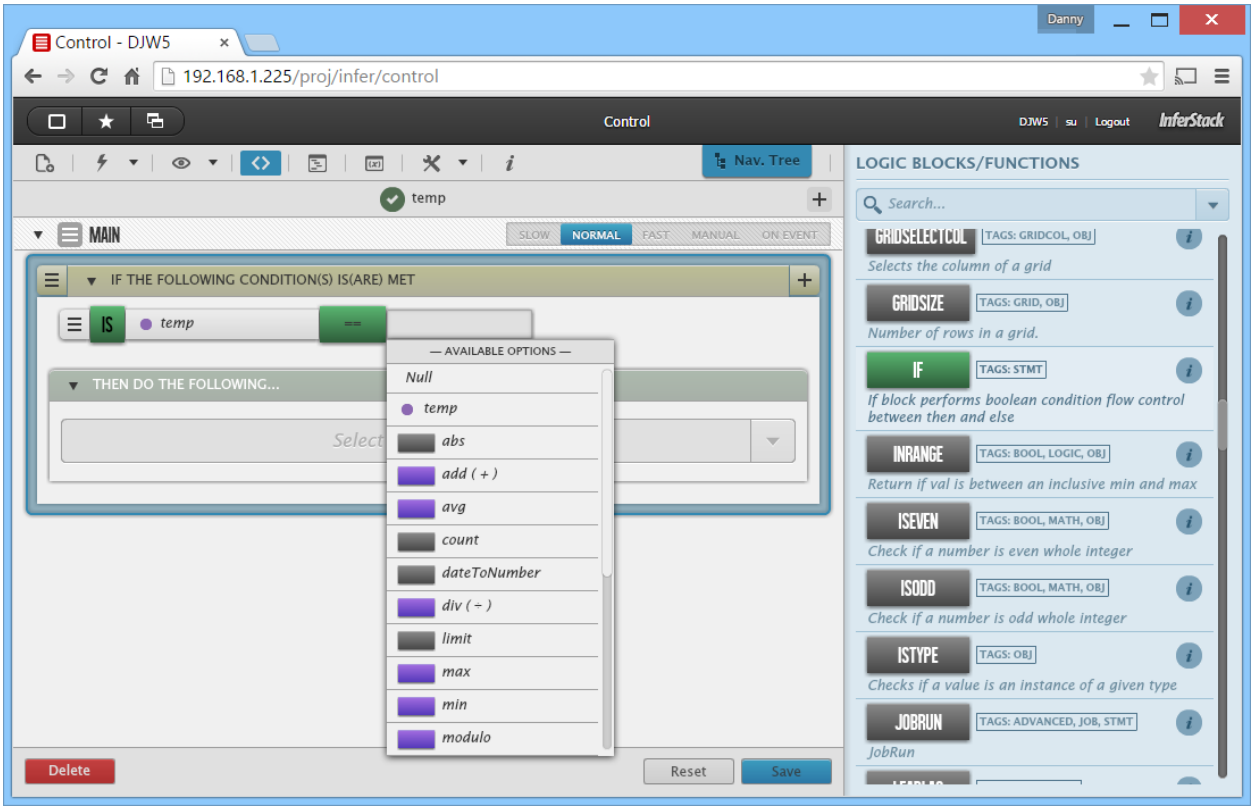
Choose a condition to evaluate.



Choose the comparison operator.

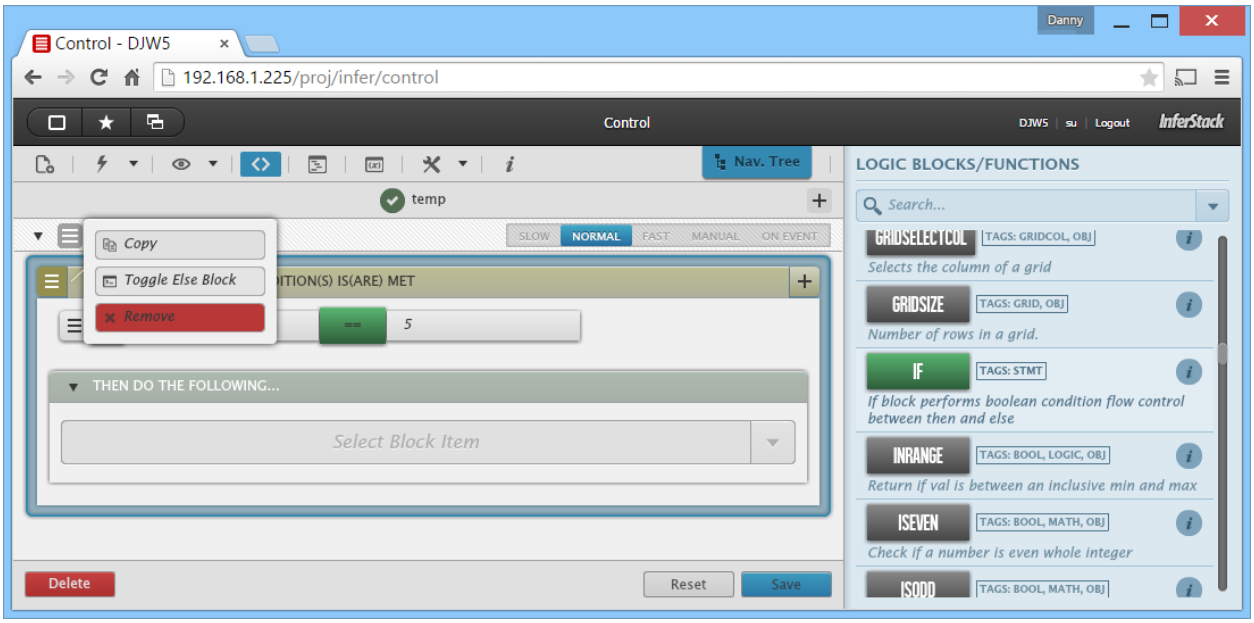


Choose a value to compare.

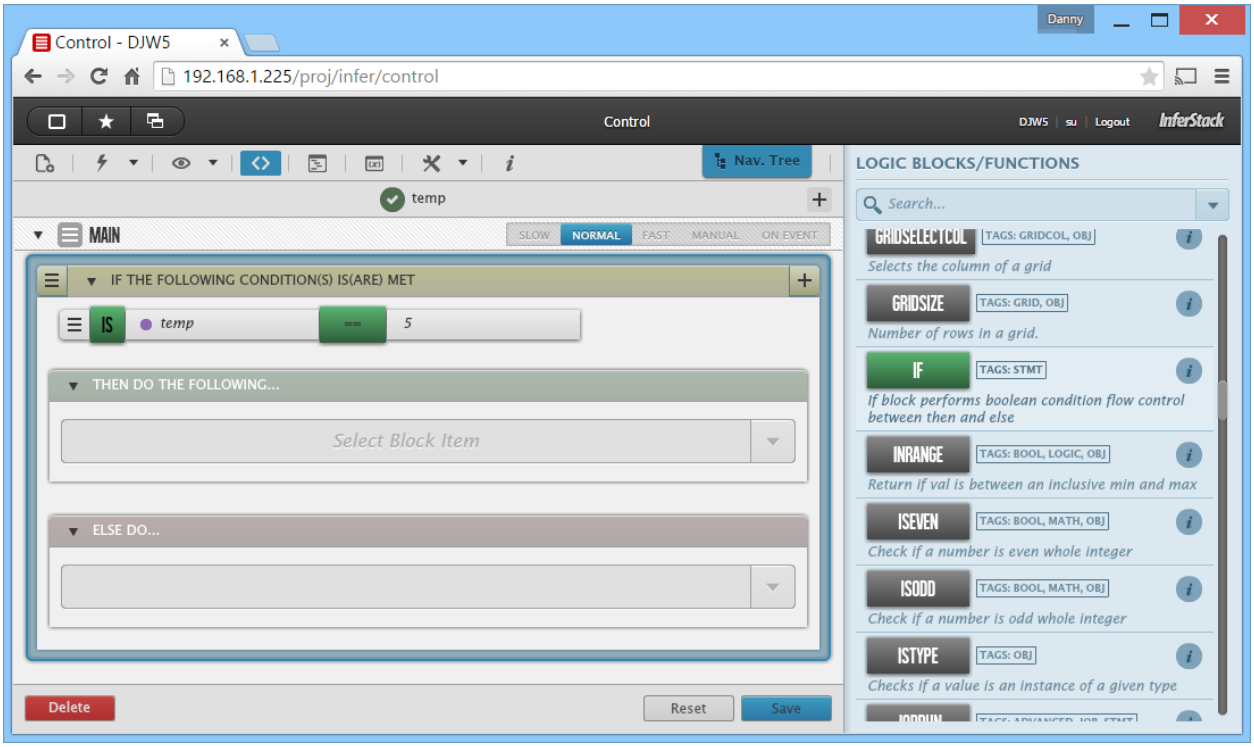


Toggle Else Block

If you want an else block (to run when the condition is not met), click the box in the upper left and choose Toggle Else Block.

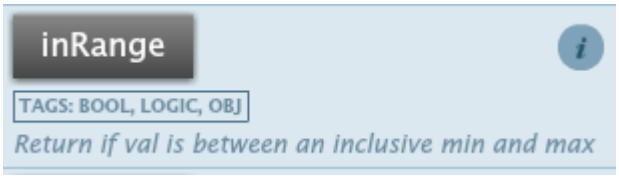


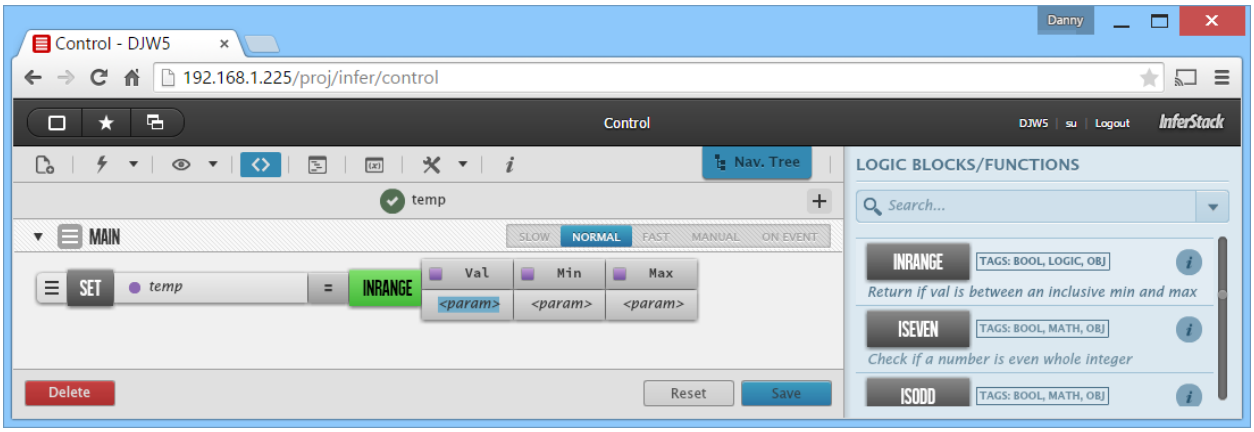
This will provide Else Do.



inRange

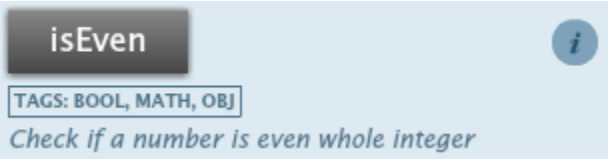
Return if val is between an inclusive min and max.

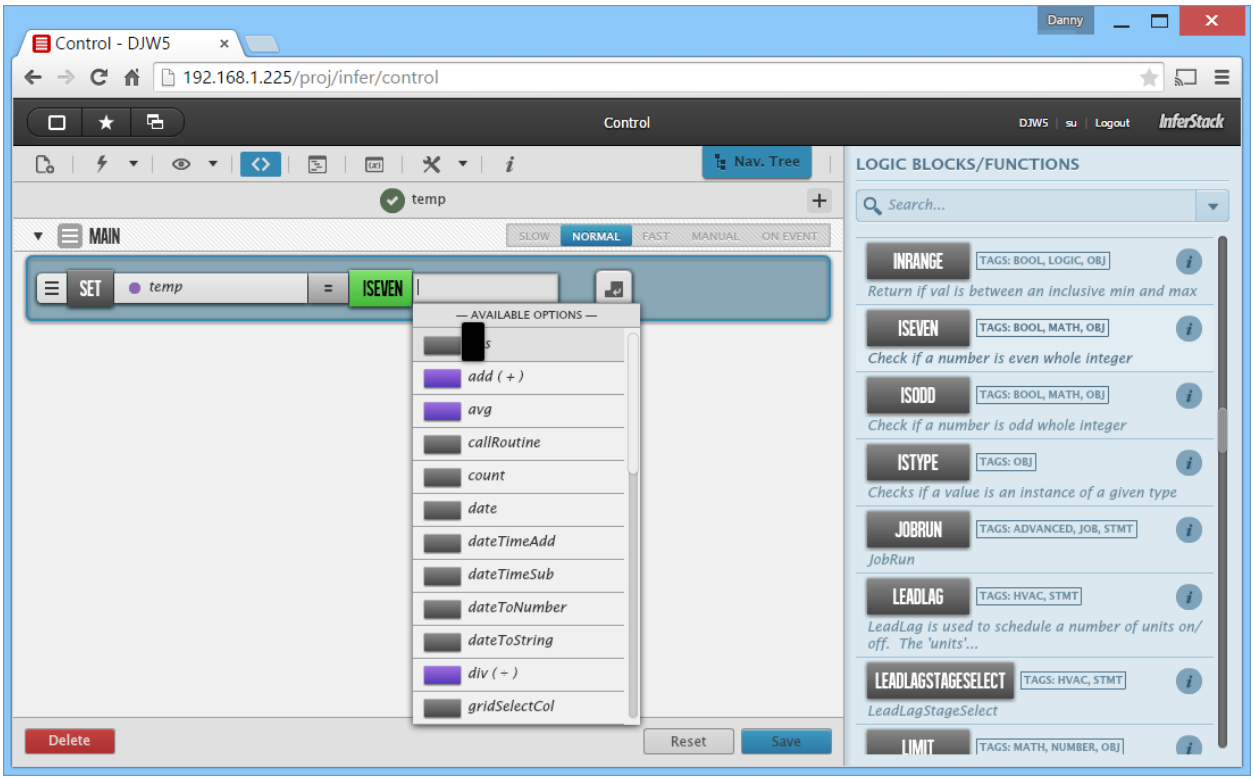




isEven

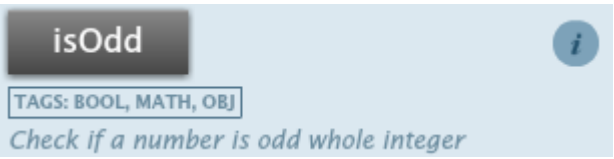
Check if a number is even whole integer.





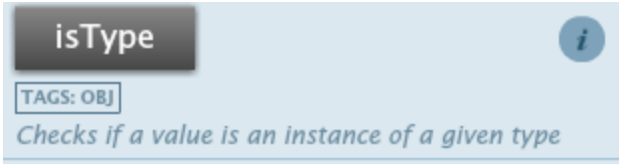
isOdd

Check if a number is odd whole integer.



isType

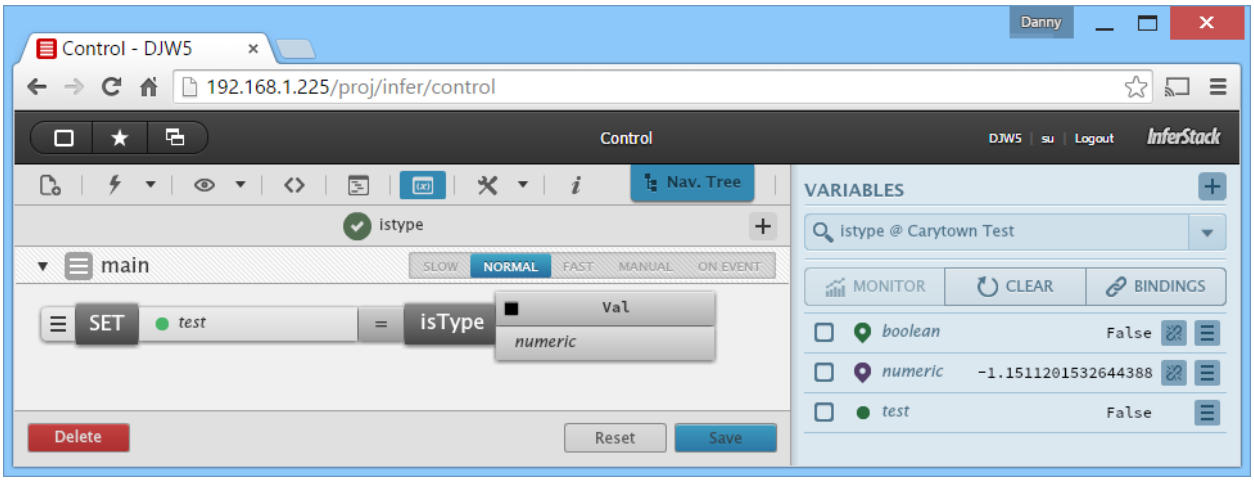
Checks if a value is an instance of a given type. Available options include Marker, Bool, Number, Str, Url, Ref, Date, Time, DateTime and Coord.



— PROPERTIES —

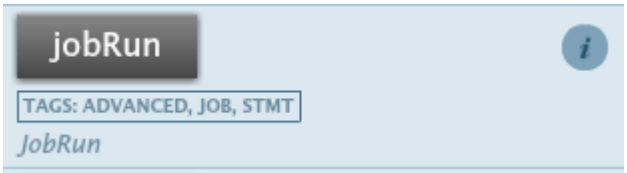
ARITY: --
Status: *ok*

NAME	TYPE	Default	TAKES	DOC
val	Block		obj	
type	Str			



jobRun

This allows you to run an axon expression or function.

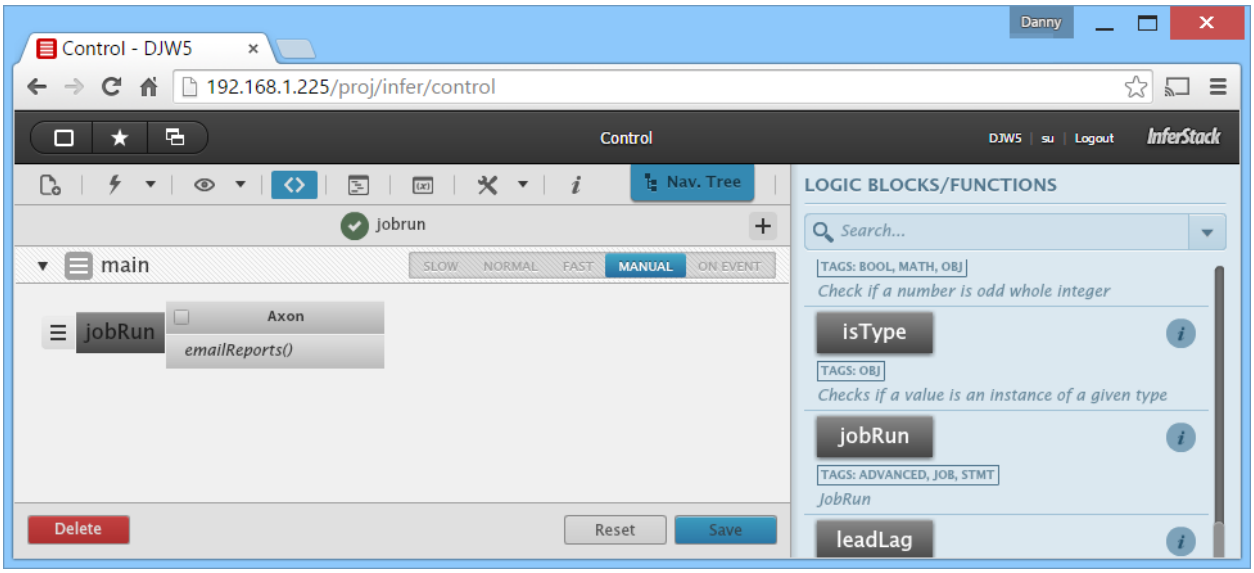


— PROPERTIES —

ARITY: --

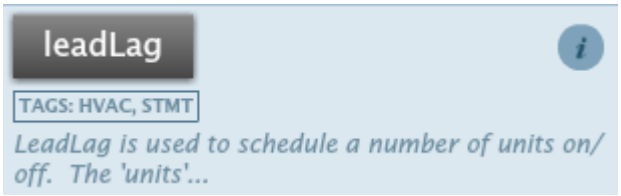
Status: *ok*

NAME	TYPE	Default	TAKES	DOC
axon	Block		str,obj	



leadLag

LeadLag is used to schedule a number of units on/off.

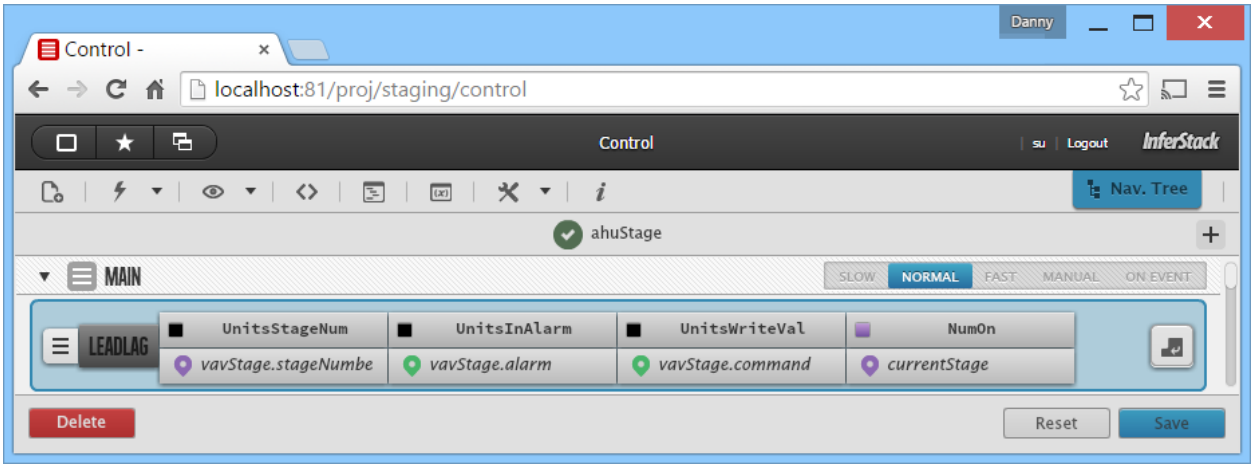


— PROPERTIES —

ARITY: --

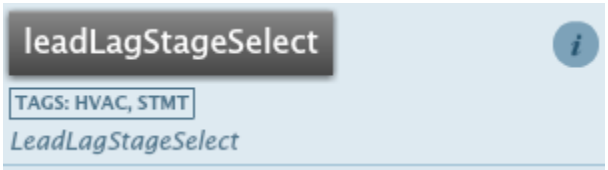
Status: *ok*

NAME	TYPE		TAKES	DOC
unitsStageNum	Block		gridCol	Units program bundle column which specifies stage n
unitsInAlarm	Block		gridCol	Units program bundle column which specifies if unit i
unitsWriteVal	Block		gridCol	Units program bundle column which specifies variable
numOn	Block		number	Number of units to command on right now



leadLagStageSelect

LeadLagStageSelect is used to select a LeadLag stage.

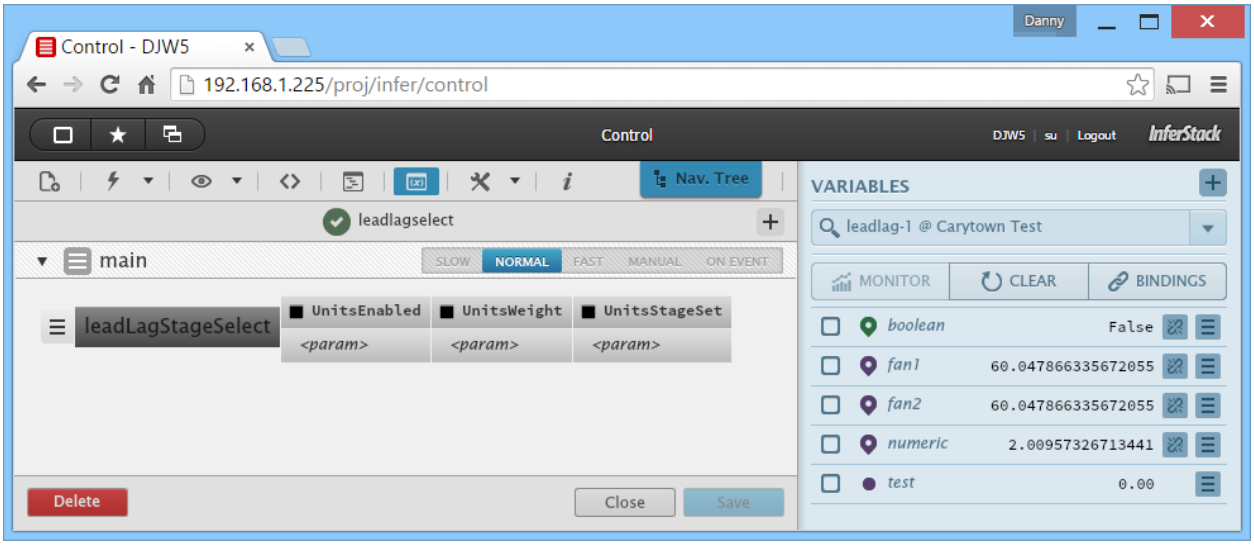


— PROPERTIES —

ARITY: --

Status: *ok*

NAME	TYPE		TAKES	DOC
unitsEnabled	Block		gridCol	Units program bundle column which specifies whether u
unitsWeight	Block		gridCol	Units program bundle column which specifies weighting
unitsStageSet	Block		gridCol	Units program bundle column which specifies variable t



limit

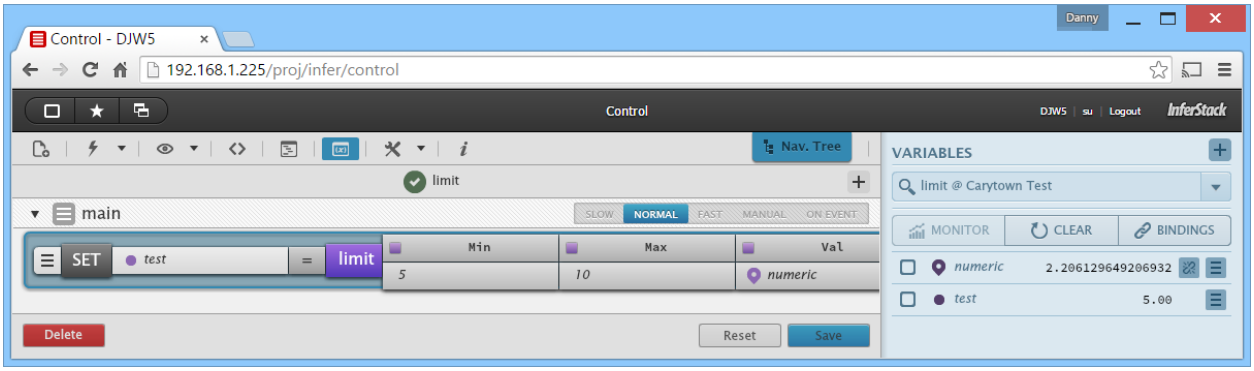
Limit val between an inclusive min and max

— PROPERTIES —

ARITY: --

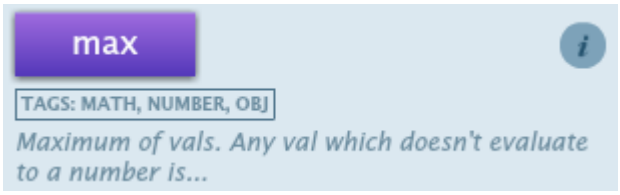
Status: *ok*

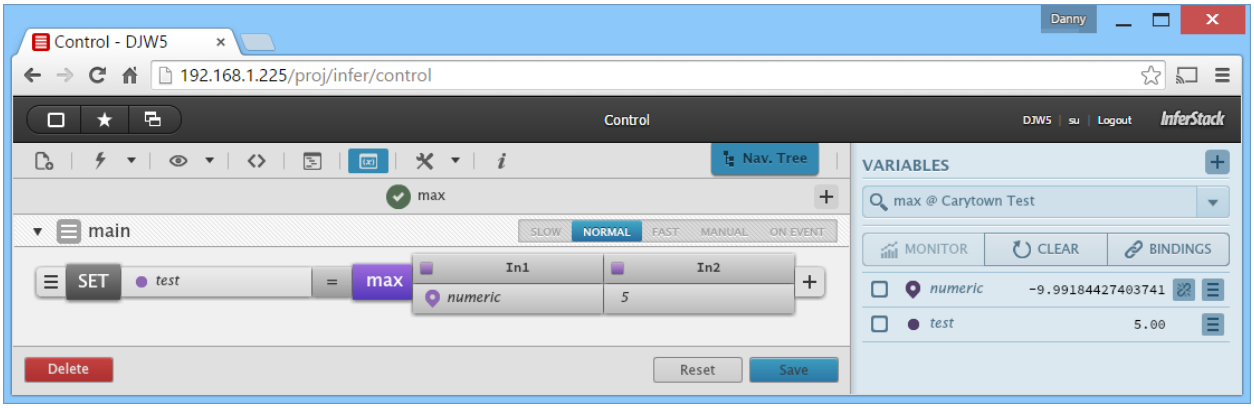
NAME	TYPE	Default	TAKES	DOC
min	Block		number,obj	
max	Block		number,obj	
val	Block		number,obj	



max

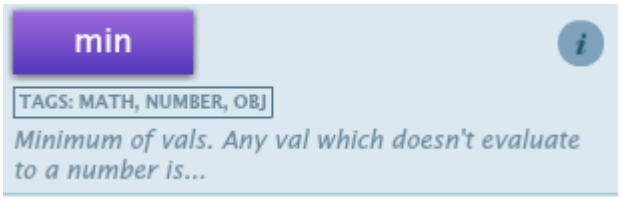
Maximum of vals. Any val which doesn't evaluate to a number is ignored. If no vals evaluate to a number then return null.

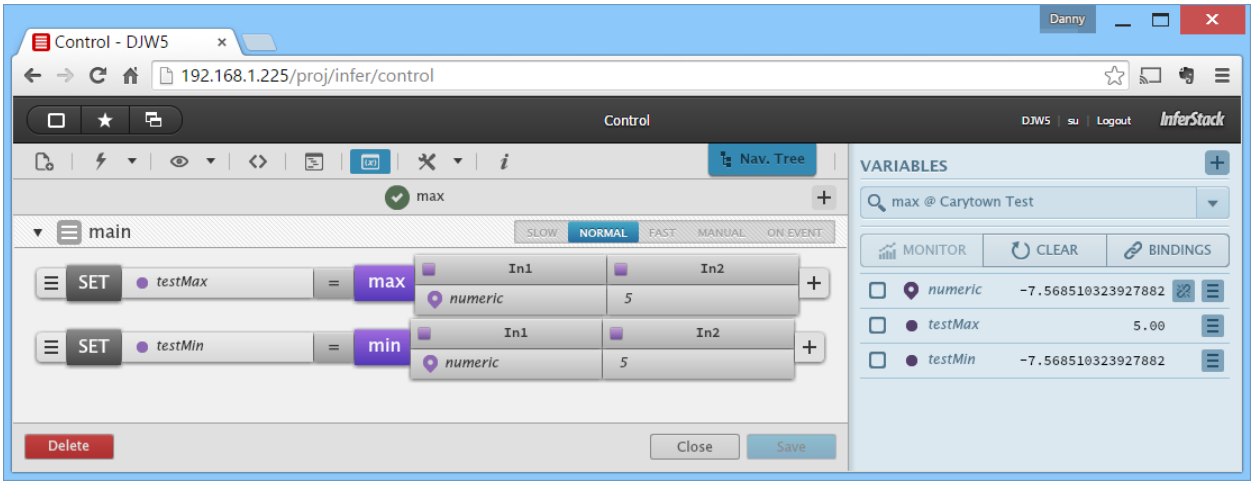




min

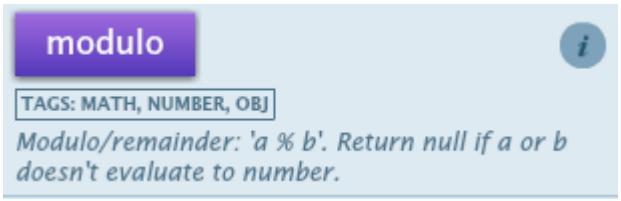
Minimum of vals. Any val which doesn't evaluate to a number is ignored. If no vals evaluate to a number then return null.





modulo

Modulo/remainder: 'a % b'. Return null if a or b doesn't evaluate to number.

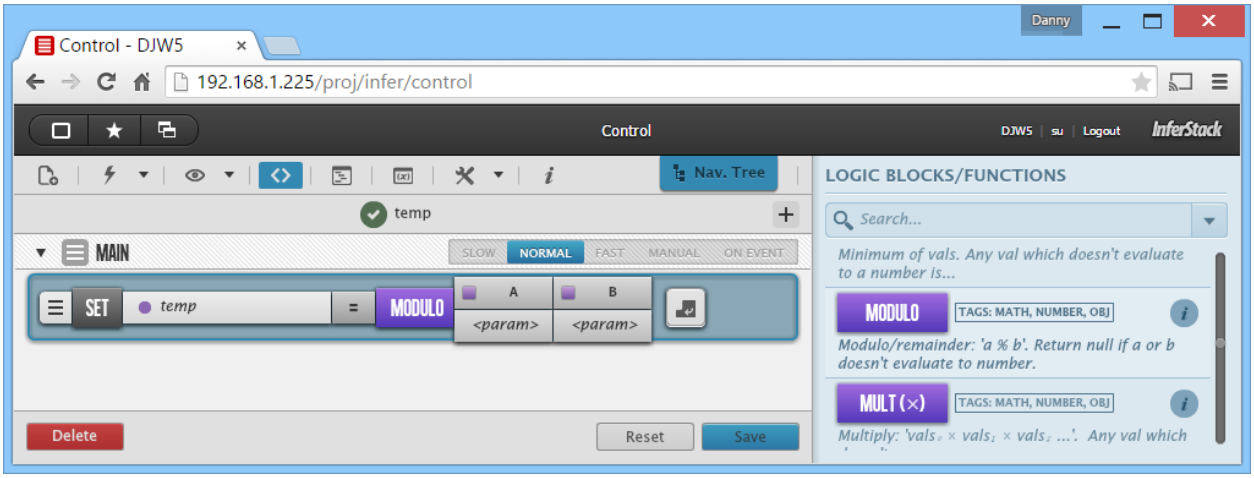


— PROPERTIES —

ARITY: *binary*

Status: *ok*

NAME	TYPE	Default	TAKES	DOC
a	Block		number,obj	
b	Block		number,obj	



mult

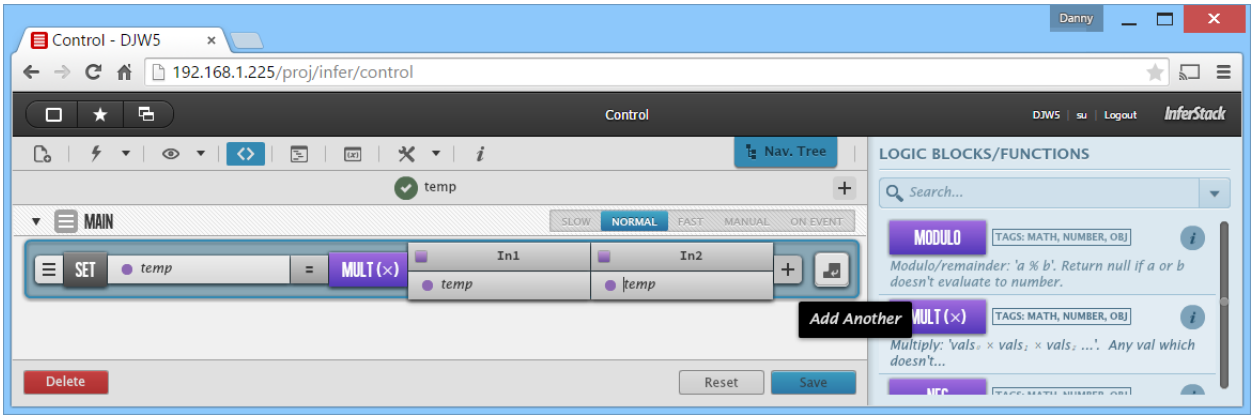
Multiply: 'vals$_0$ × vals$_1$ × vals$_2$ ...'. Any val which doesn't evaluate to a number is ignored. If no vals evaluate to a number then return null.

— PROPERTIES —

ARITY: *n*

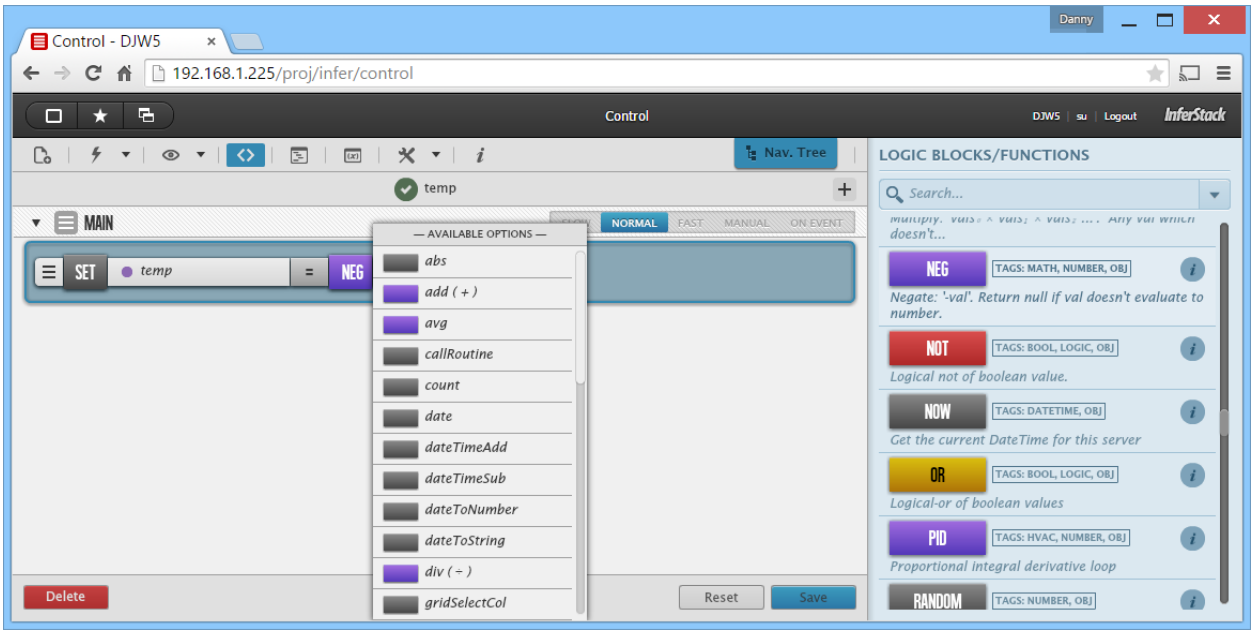
Status: *ok*

NAME	TYPE	Default	TAKES	DOC
vals	Block		number,gridCol,obj	



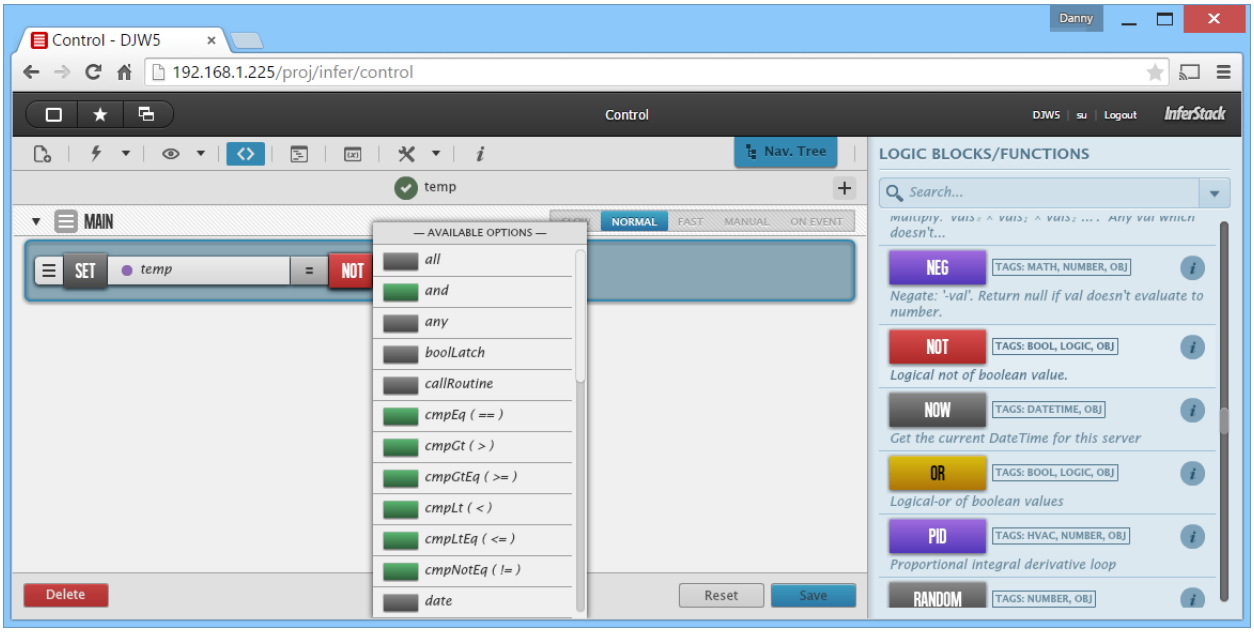
neg

Negate: '-val'. Return null if val doesn't evaluate to number.



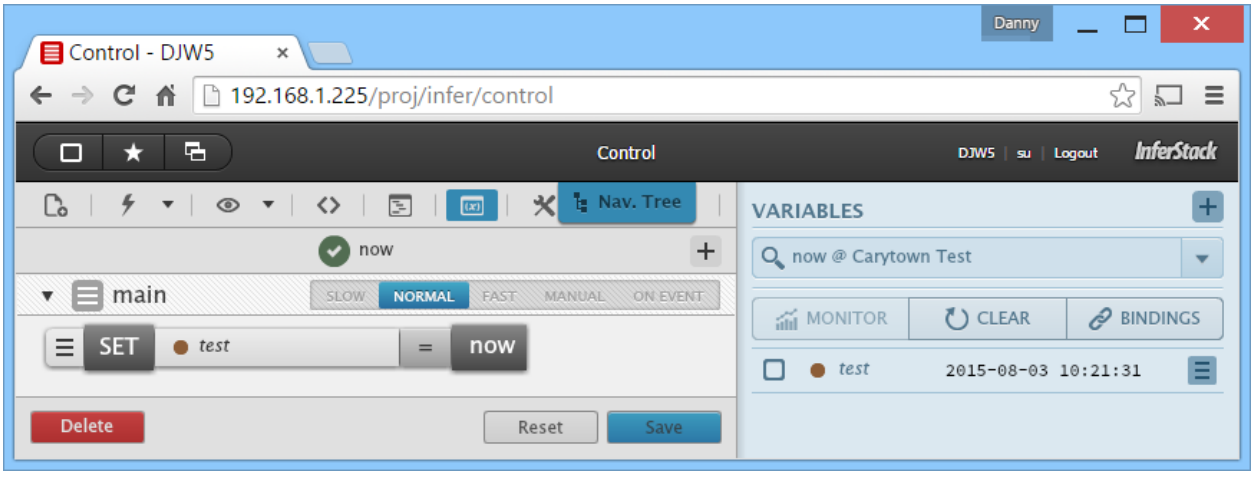
not

Logical not of boolean value.



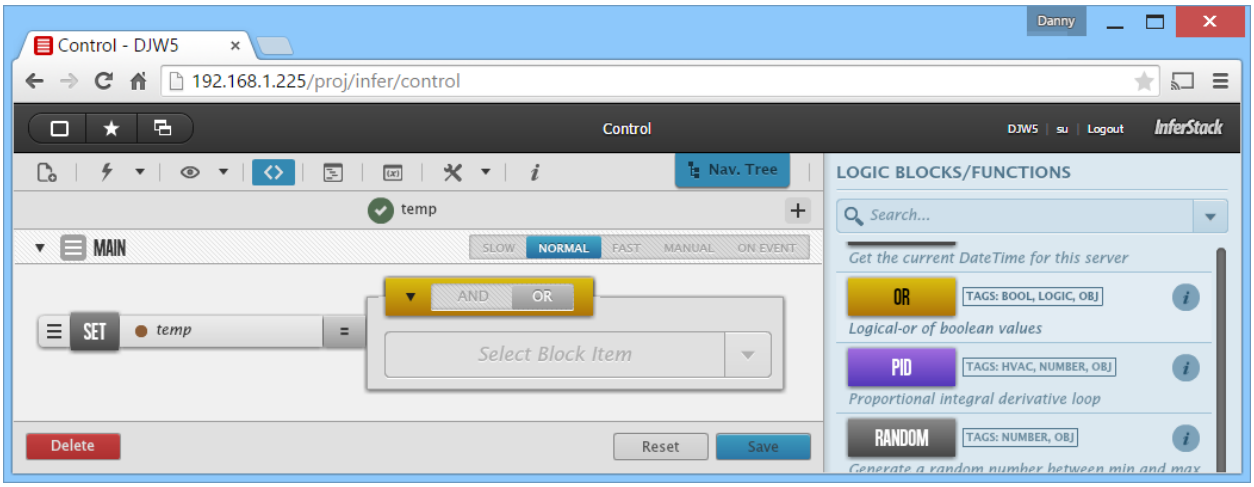
now

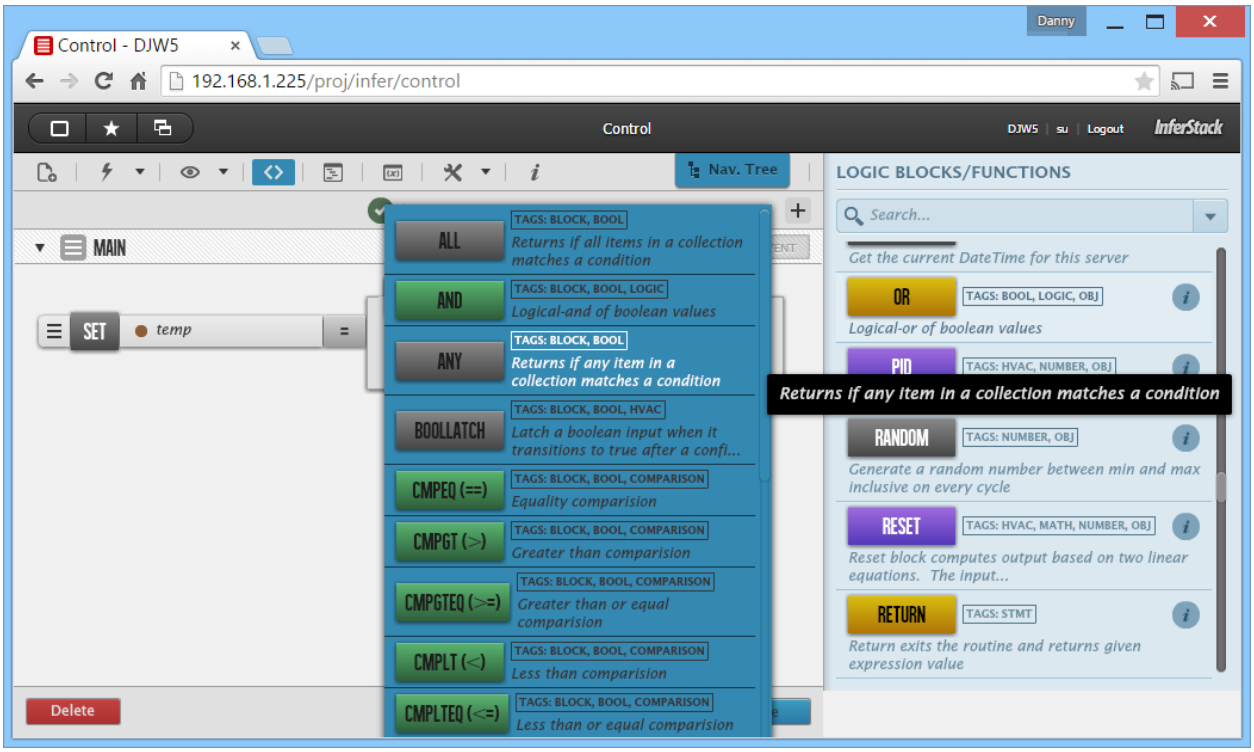
Get the current DateTime for this server.



or

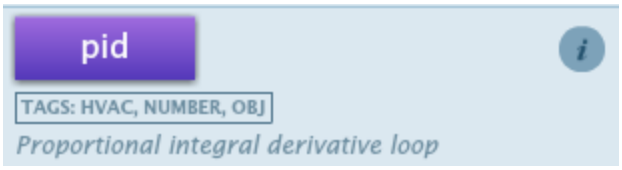
Logical-or of boolean values





PID

Proportional integral derivative loop

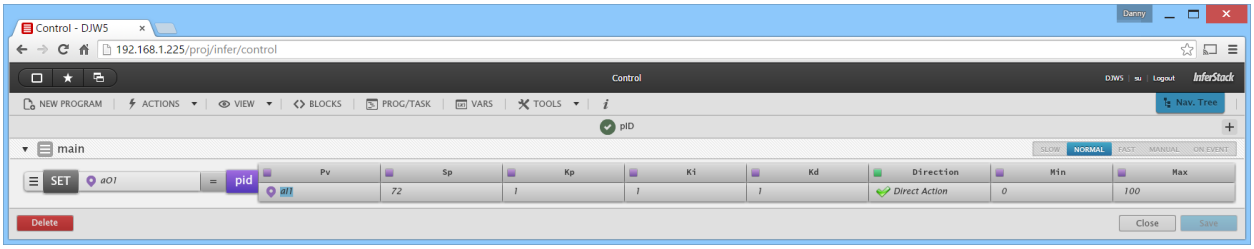


— PROPERTIES —

ARITY: --

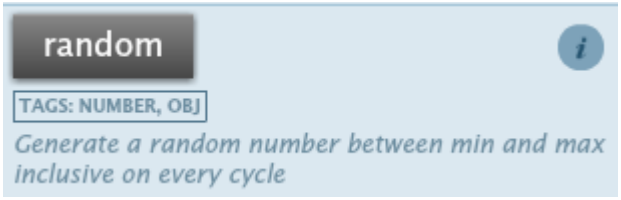
Status: *ok*

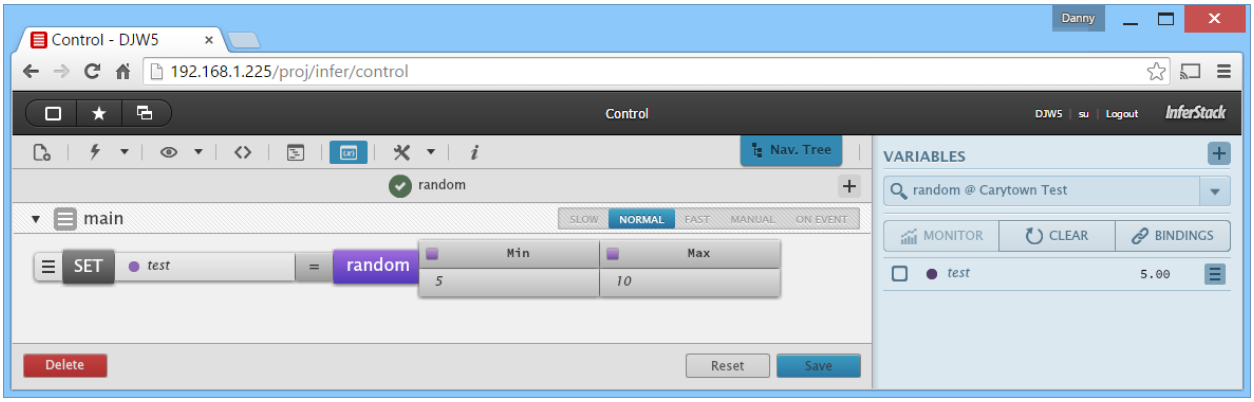
NAME	TYPE	Def...	TAKES	DOC
pv	Block		number,obj	Process variable measuring process
sp	Block		number,obj	Setpoint to compare process variable
kp	Block		number,obj	Proportional term - band/gain
ki	Block		number,obj	Integral term - gain/reset
kd	Block		number,obj	Derivative term - gain/rate
directior	Block		bool,obj	Direction of loop
min	Block		number,obj	Min value to clip output
max	Block		number,obj	Max value to clip output



random

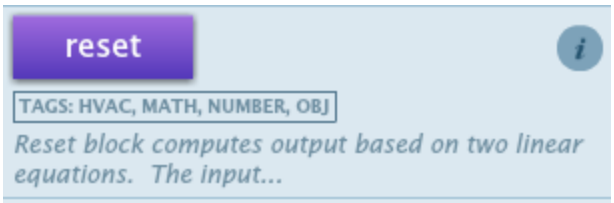
Generate a random number between min and max inclusive on every cycle





reset

Reset block computes output based on two linear equations. The input is clipped to inLo and inHi,the percentage of input along the linear segment between inLo and inHi is computed,and then that percentage is output as a Number between outLo and outHi.

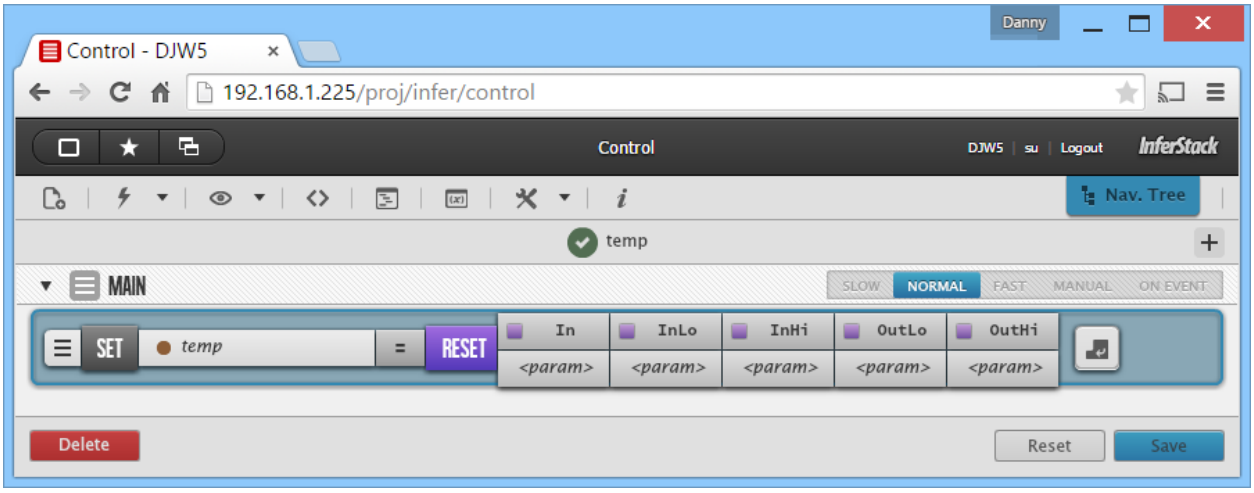


— PROPERTIES —

ARITY: --

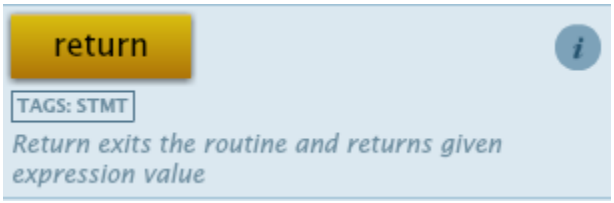
Status: *ok*

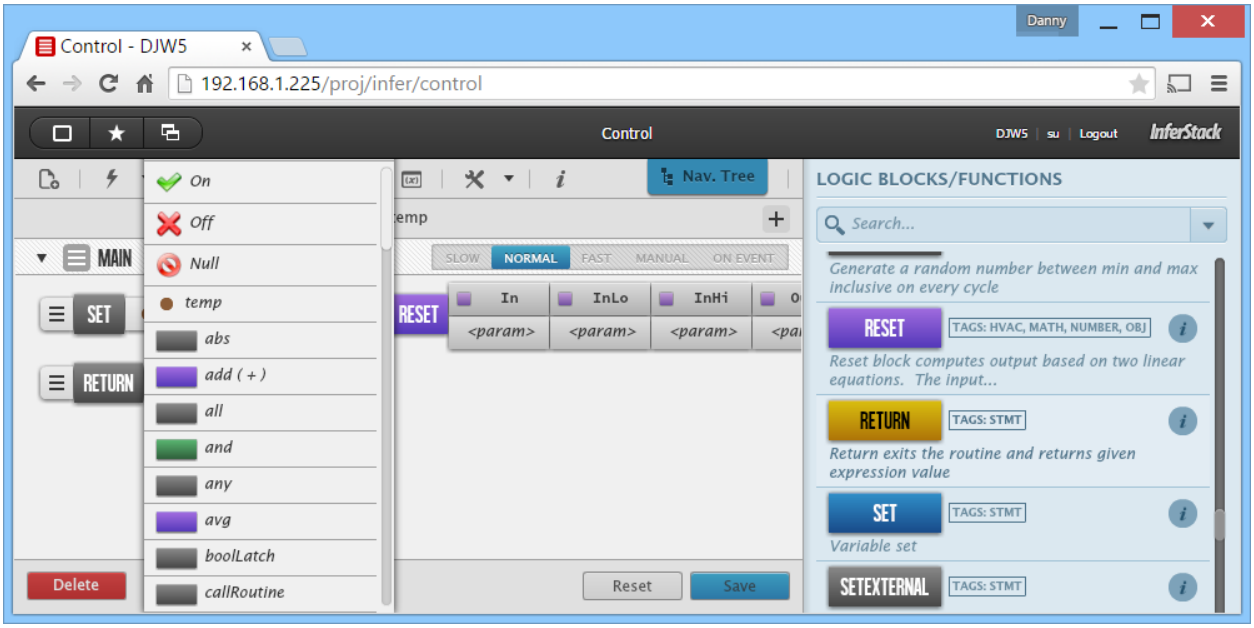
NAME	TYPE	Def...	TAKES	DOC
in	Block		number,obj	Input variable used to compute output
inLo	Block		number,obj	Low value of the input used to compute inp This value must be less than inHi.
inHi	Block		number,obj	High value of the input used to compute in This value must be greater than inLo.
outLo	Block		number,obj	Low value of the output used to compute o If this value is greater than outHi, the reset
outHi	Block		number,obj	High value of the output used to compute (If this value is less than outLo, the reset is



return

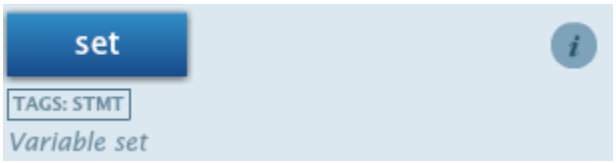
Return exits the routine and returns given expression value

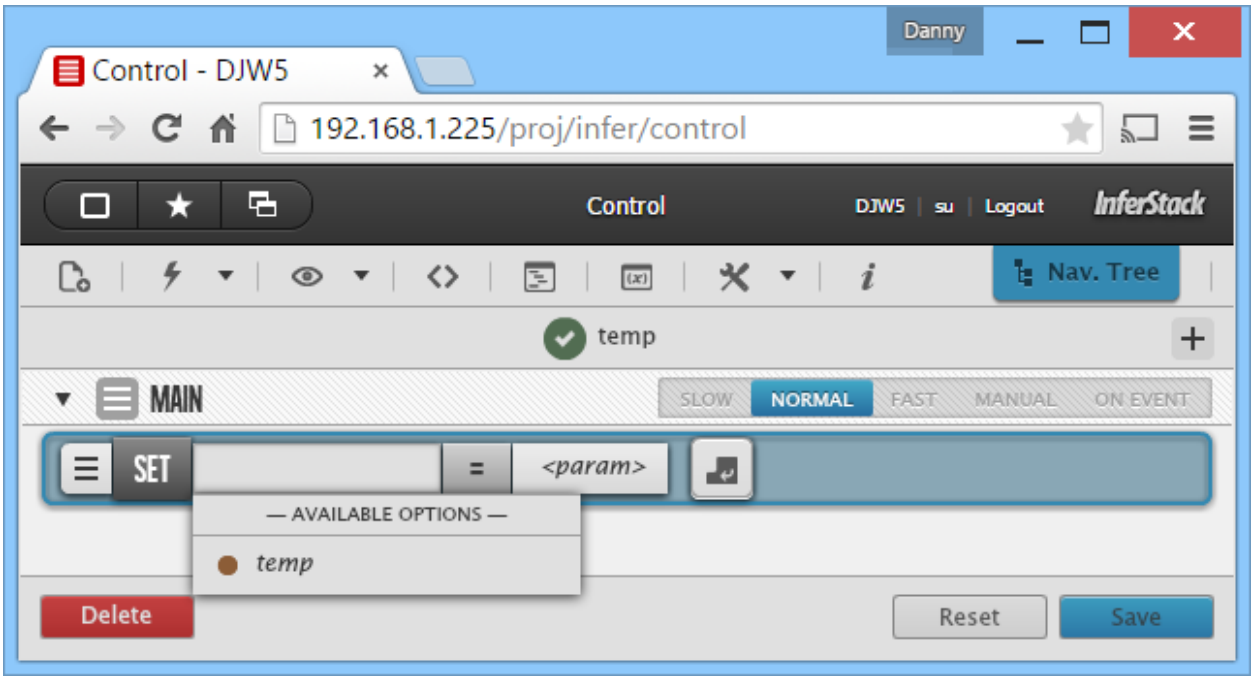


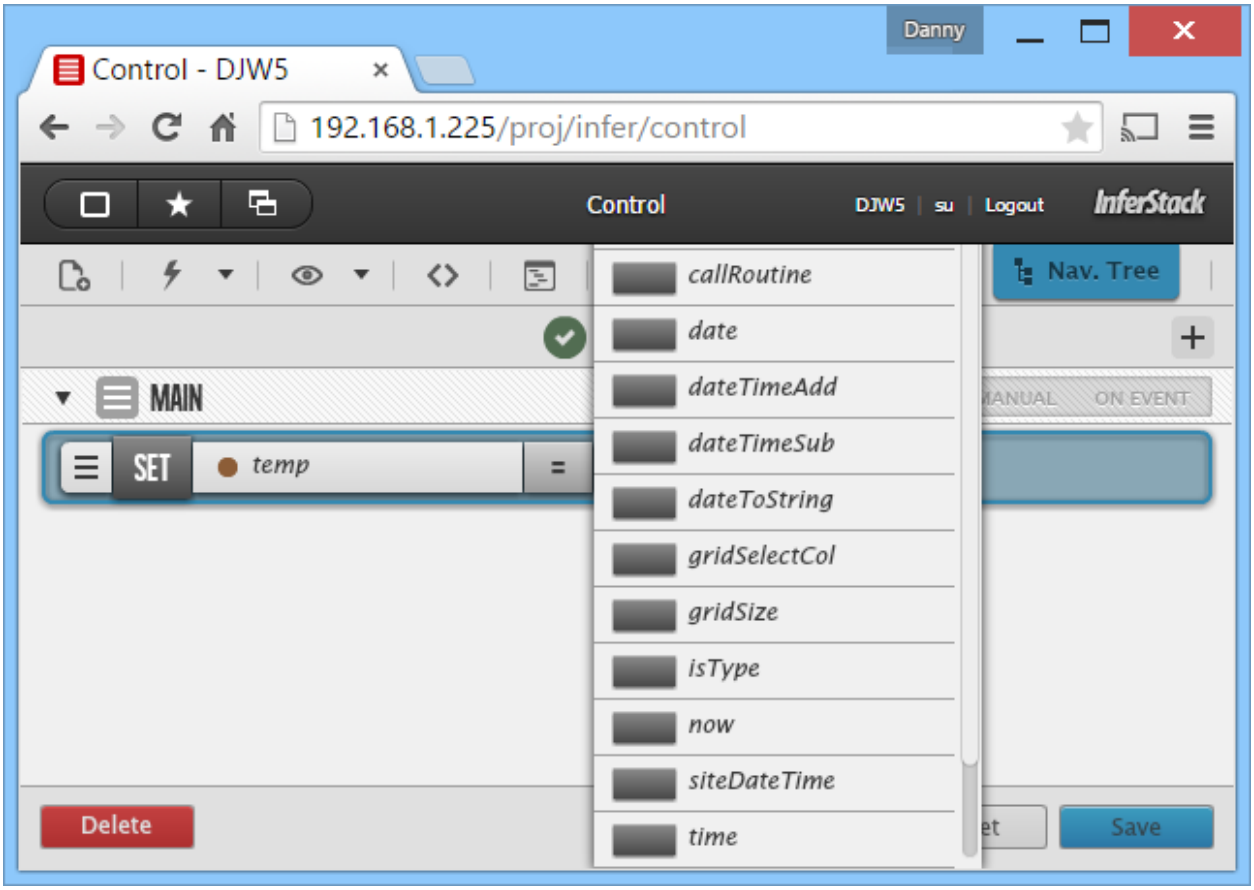


set

Variable set

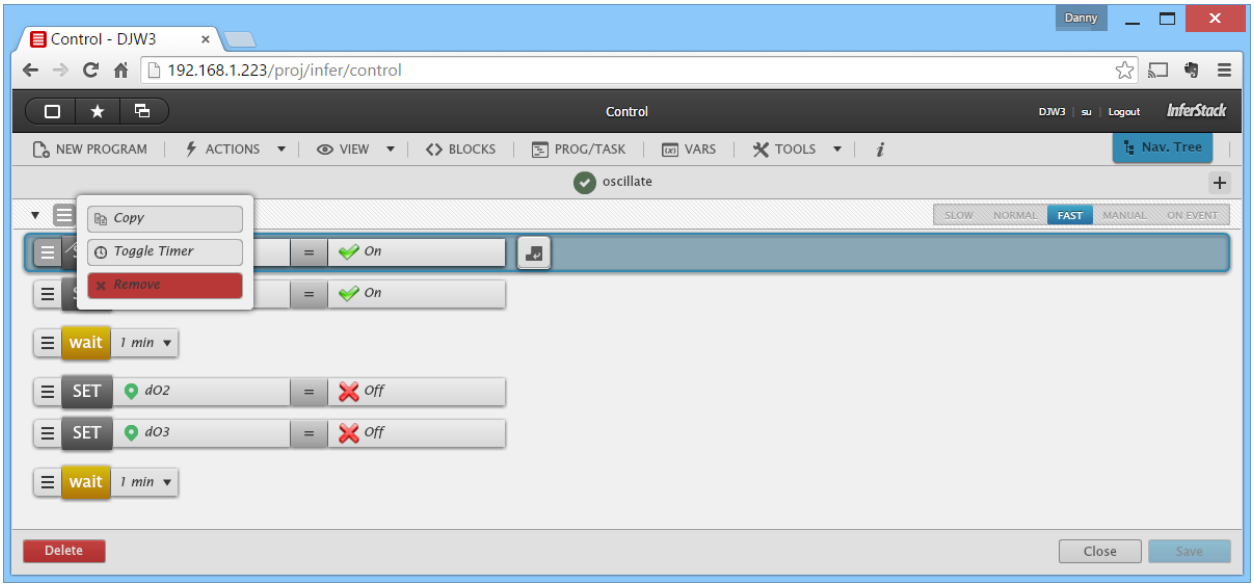

set
TAGS: STMT
Variable set

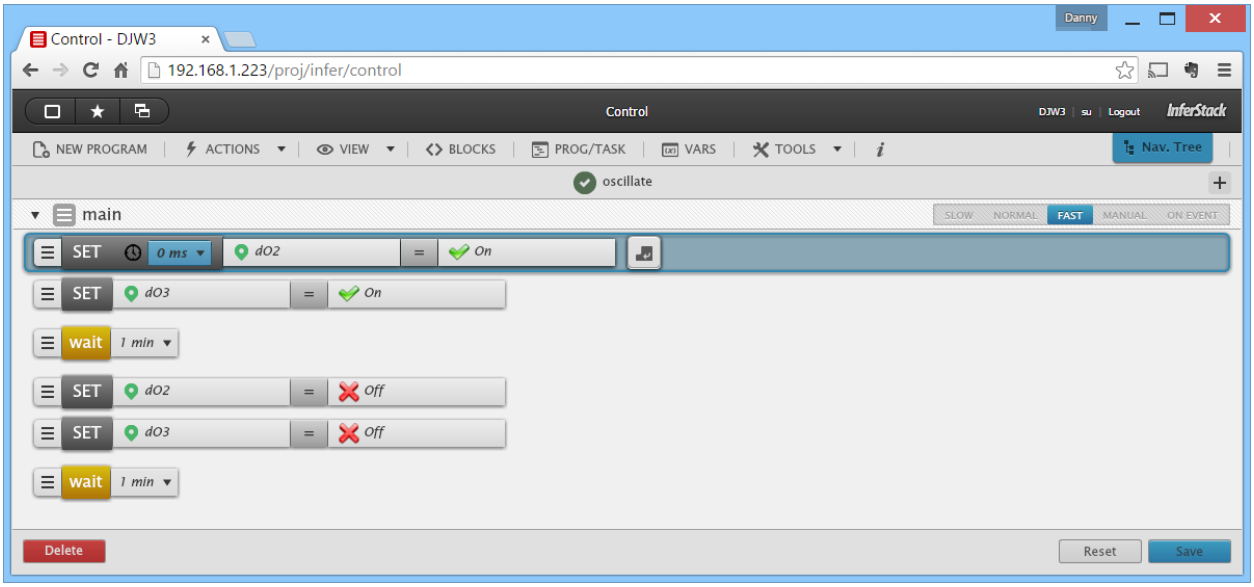

Danny
Control - DJW5
192.168.1.225/proj/infer/control
Control
DJW5 | su | Logout
InferStack
Nav. Tree
temp
MAIN
SLOW
NORMAL
FAST
MANUAL
ON EVENT
SET
=
<param>
— AVAILABLE OPTIONS —
temp
Delete
Reset
Save



Toggle Timer

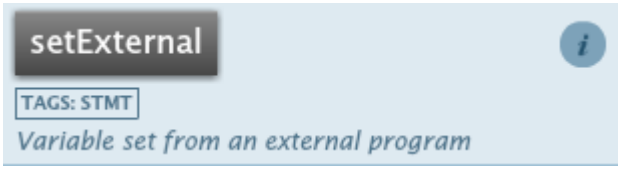
Toggle Timer is available on the Set block and the timer just delays the output for that time delay. For example, if it is set to 5 seconds, then when the output has a transition it just waits 5 seconds before sending it. If the variable is a number, let's say 45, and the code then changes it to 90, it will wait 5 seconds before making the value 90.

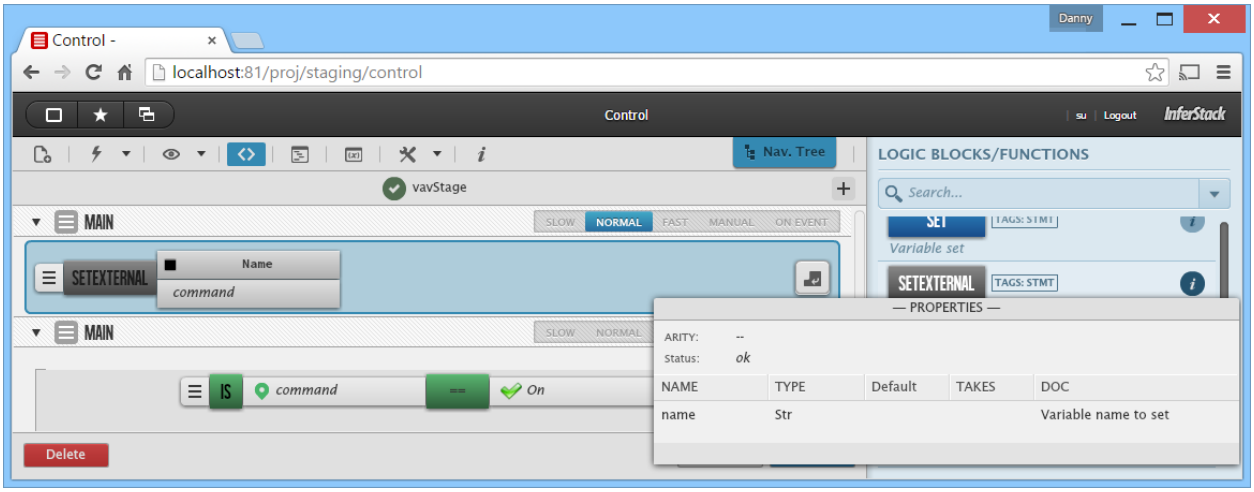




setExternal

Variable set from an external program like setting command in the example below.





sineWave

Generate a sine wave

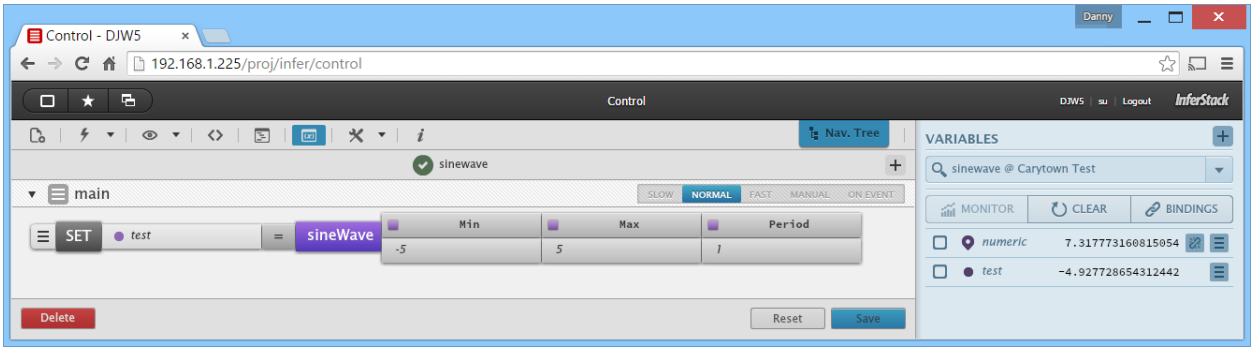
Note: This does not work until 2.1.11.

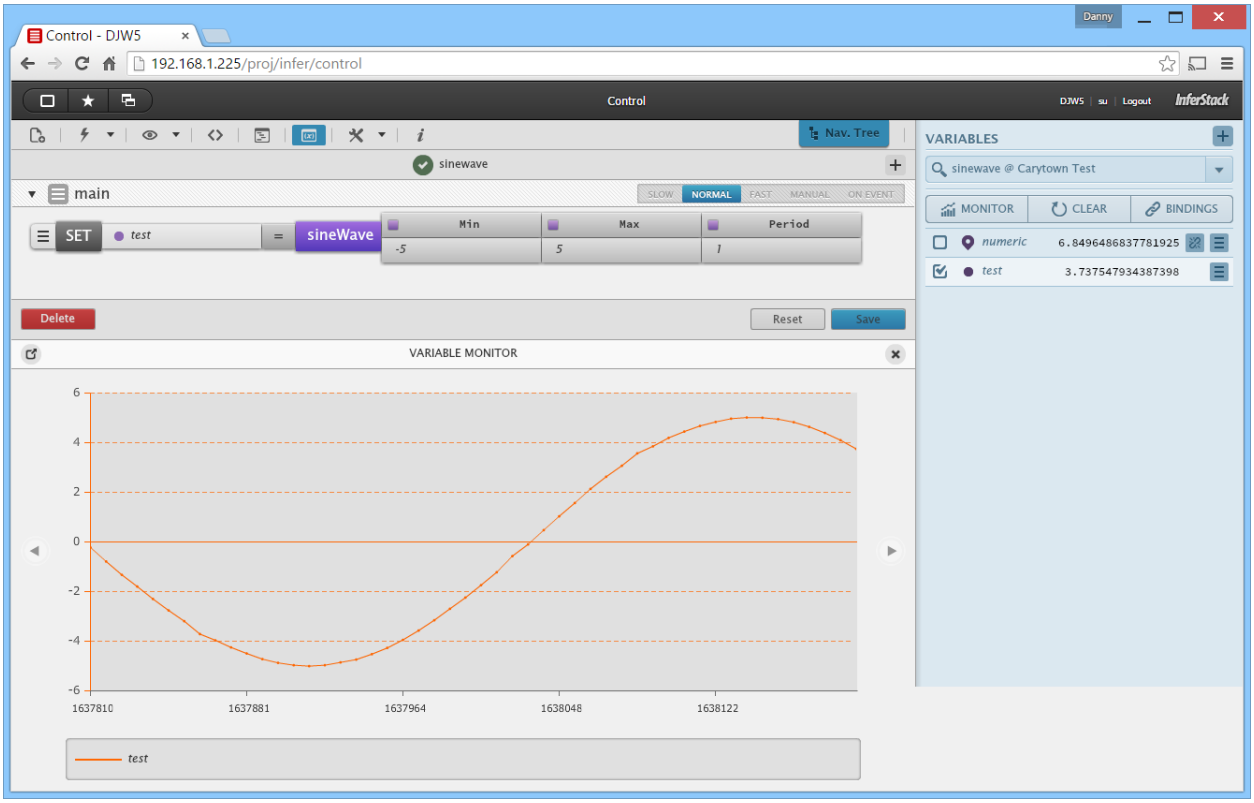
— PROPERTIES —

ARITY: --

Status: *ok*

NAME	TYPE	Default	TAKES	DOC
min	Block		number,obj	Min value of the range
max	Block		number,obj	Max value of the range
period	Block		number,obj	Period for a full cycle

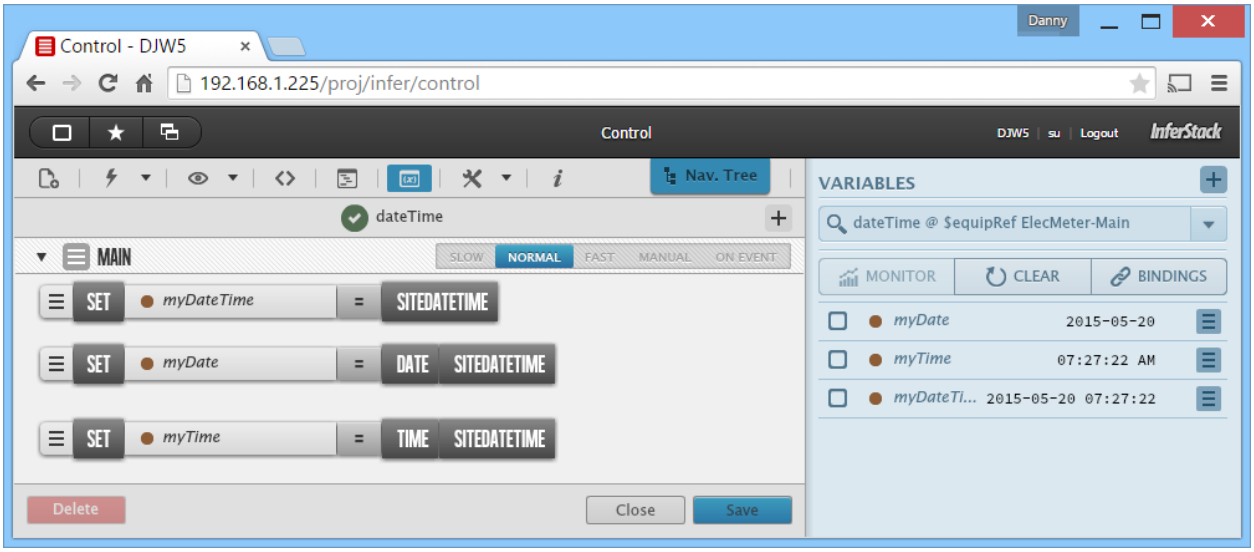




siteDateTime

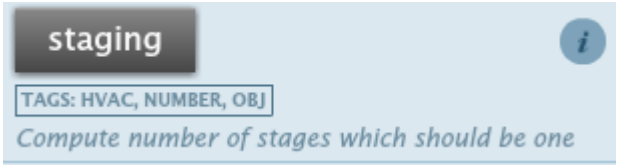
Get the current dateTime for the site associated with the target.

You can use this to get date, time and dateTime.



staging

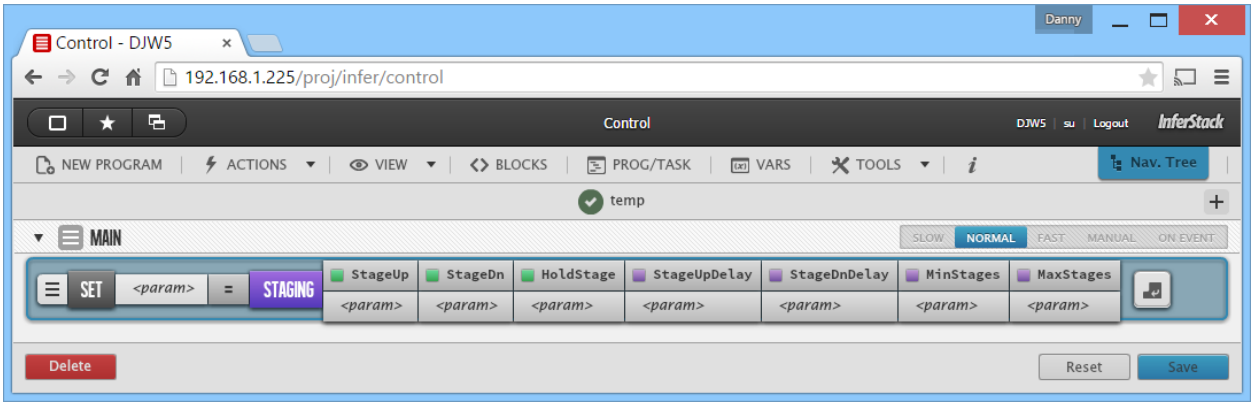
Compute the number of stages which should be on.



— PROPERTIES —

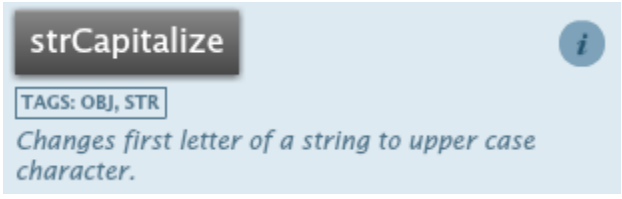
ARITY: --
Status: *ok*

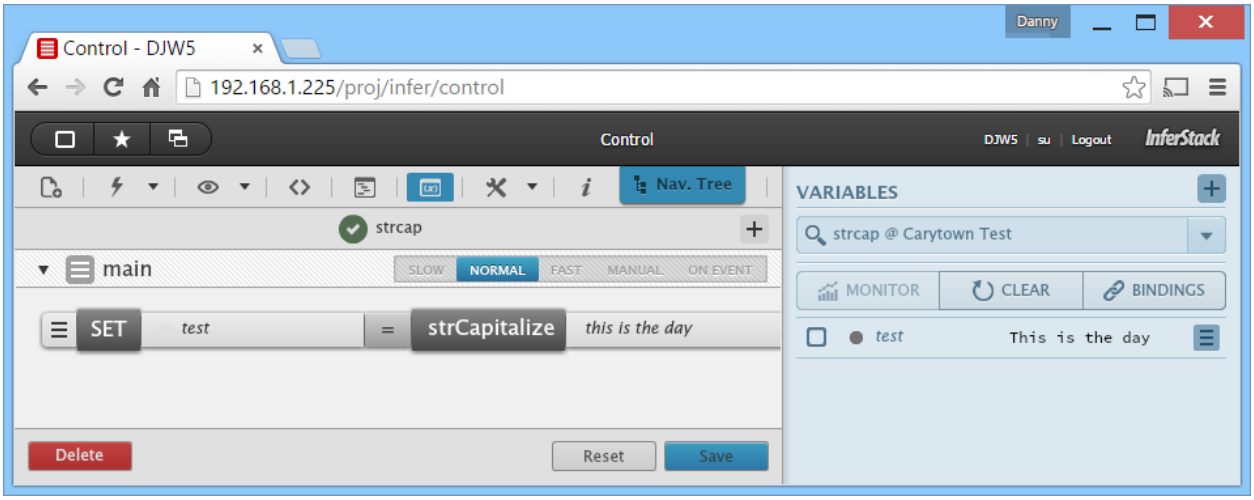
NAME	TYPE	Default	TAKES	DOC
stageUp	Block		bool,obj	Stage up
stageDn	Block		bool,obj	Stage down
holdStage	Block		bool,obj	Hold current numb
stageUpDelay	Block		number,obj	Duration number f
stageDnDelay	Block		number,obj	Duration number f
minStages	Block		number,obj	Min number of stag
maxStages	Block		number,obj	Max number of sta



strCapitalize

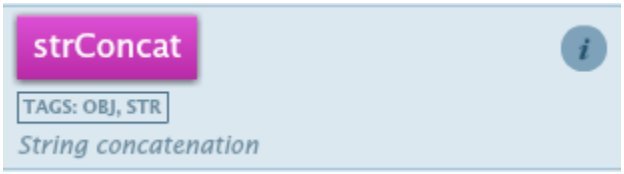
Changes first letter of a string to upper case character.

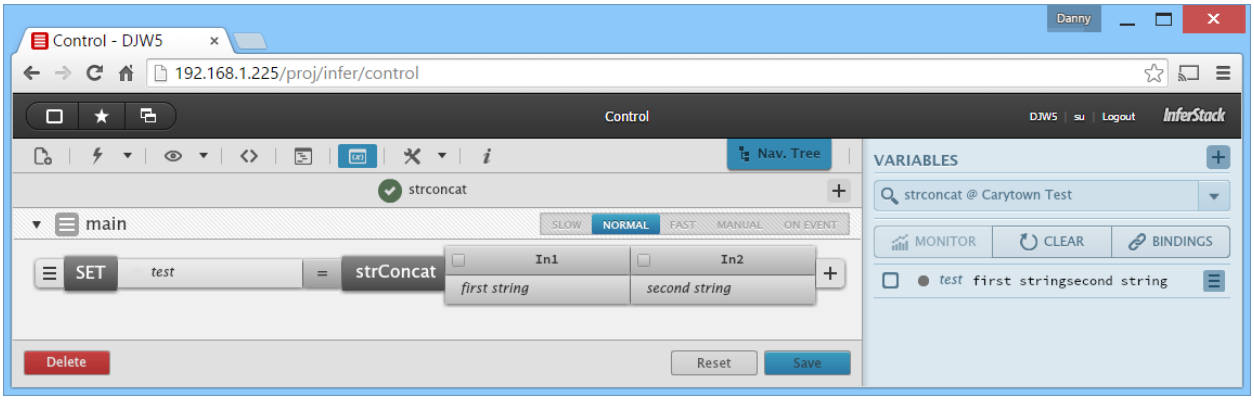




strConcat

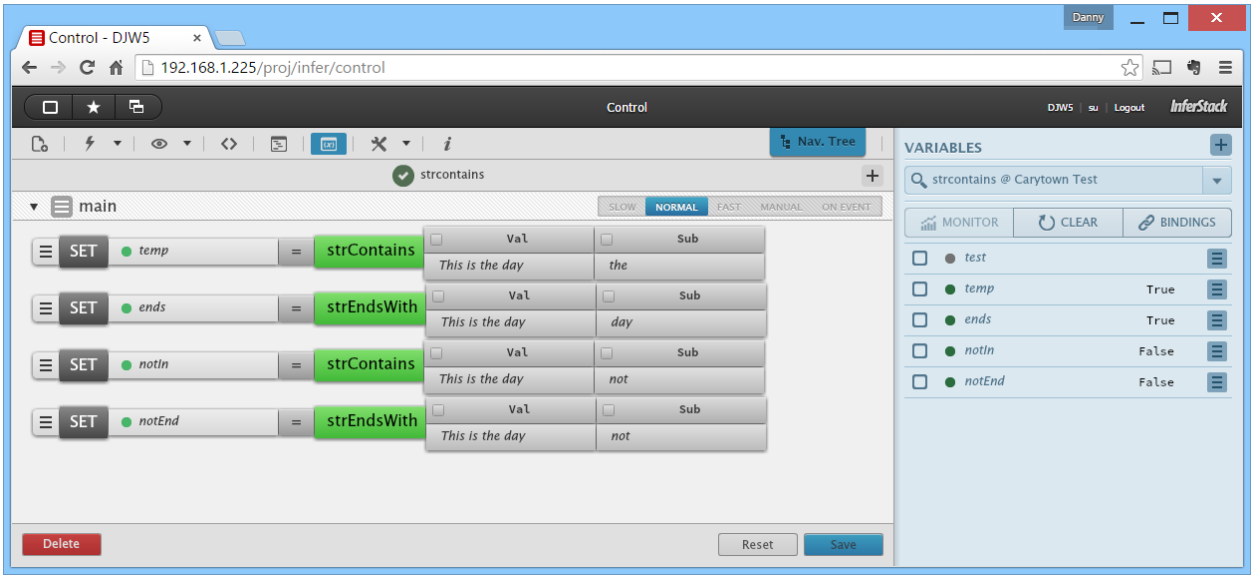
String concatenation





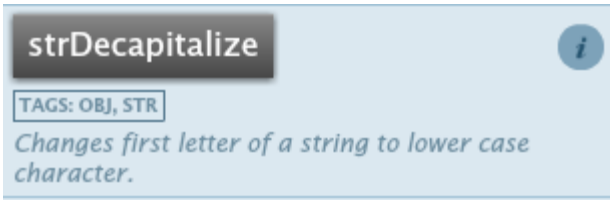
strContains

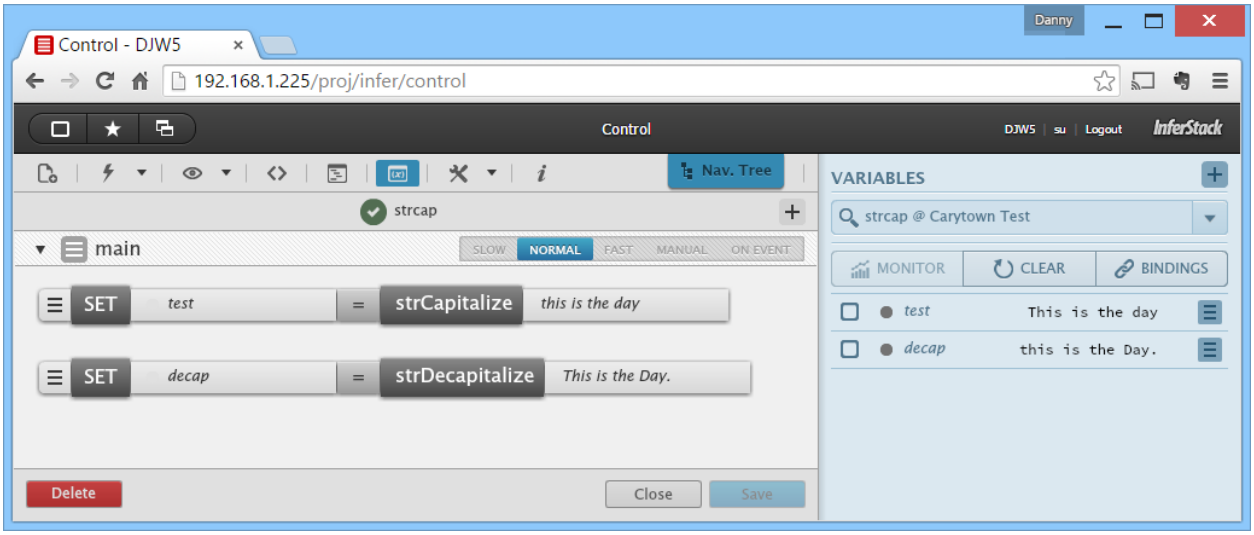
Compares if a val string contains substring.



strDecapitalize

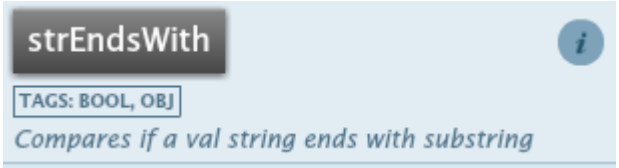
Changes first letter of a string to lower case character.

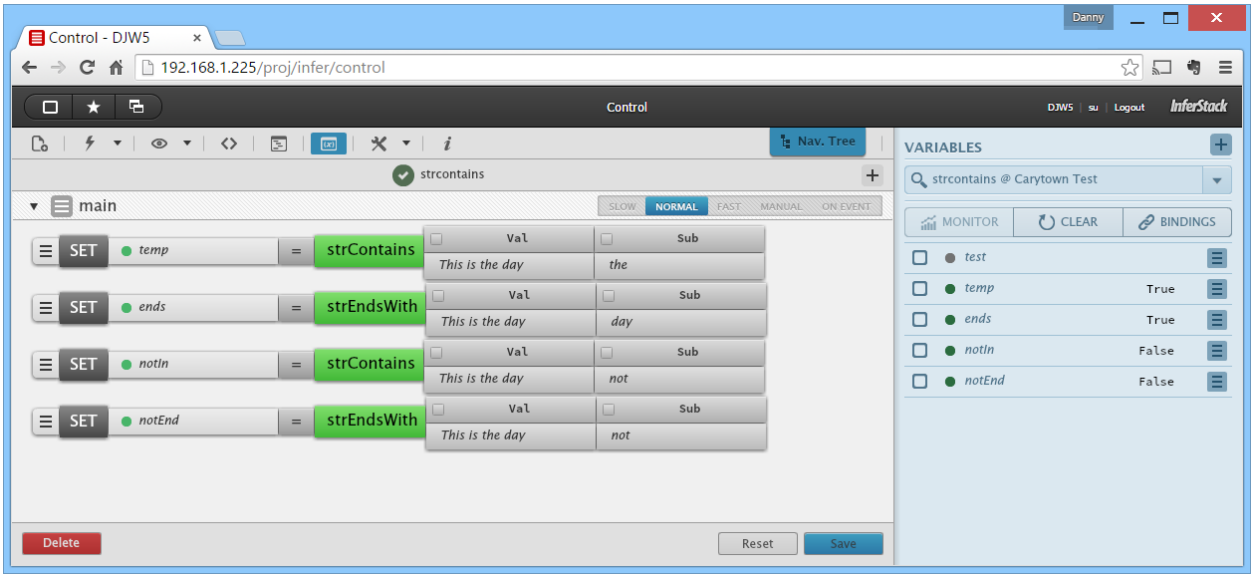




strEndsWith

Compares if a val string ends with substring.





strFormat

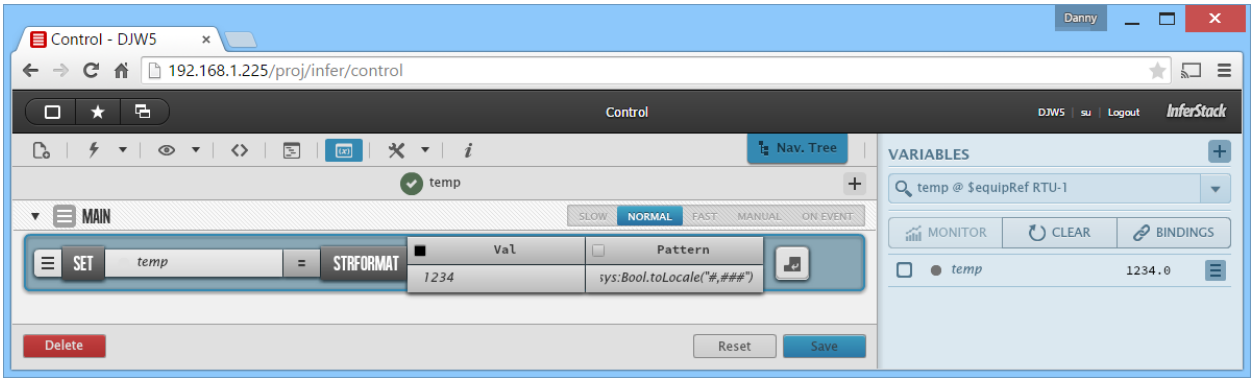
String format - see `format()`

— PROPERTIES —

ARITY: --

Status: *ok*

NAME	TYPE	Default	TAKES	DOC
val	Block		obj	Value to format
pattern	Block		str,obj	Pattern to use for formatti - `sys::Bool.toLocale`



Format()

```
format(val, pattern: "")
```

Format an object using the current locale and specified format pattern. Formatting patterns follow Fantom toLocale conventions:

- `Bool.toLocale` - see /doc/sys/Bool.html#toLocale
- `Int.toLocale` - see /doc/sys/Int.html#toLocale
- `Float.toLocale` - see /doc/sys/Float.html#toLocale
- `Duration.toLocale` - see /doc/sys/Duration.html#toLocale
- `Date.toLocale` - see /doc/sys/Date.html#toLocale
- `Time.toLocale` - see /doc/sys/Time.html#toLocale
- `DateTime.toLocale` - see /doc/sys/DateTime.html#toLocale

```
Str toLocale()
```

Return `toStr` - enabled duck typing for `toLocale`.

Bool.toLocale

Return localized strings for "true" and "false" using current locale.

Int.toLocale

```
Str toLocale(Str? pattern := null)
```

Format this integer number for the current locale. If pattern is null, then the locale's default pattern is used. See `Float.toLocale` for pattern language. Fractional formatting is not supported for integers.

In addition `Int.toLocale` supports the "B" pattern which will format a number of bytes with the appropiate B, KB, MB, GB suffix based on the magnitide (1024B == 1KB).

Examples:

```
3.toLocale("00")              =>  03
3.toLocale("000")             =>  003
123456789.toLocale("#,###")   =>  123,456,789
123.toLocale("B")             =>  123B
1234.toLocale("B")            =>  1.2KB
100_000.toLocale("B")         =>  98KB
(3*1024*1024).toLocale("B")   =>  3MB
```

Float.toLocale

```
Str toLocale(Str? pattern := null)
```

Format this floating point number for the current locale. If pattern is null, then the locale's default pattern is used. Also see `Num.localeDecimal`, `Num.localeGrouping`, etc.

The pattern format:

```
#   optional digit
0   required digit
.   decimal point
,   grouping separator (only last one before decimal matters)
```

Examples:

```
12345.786f.toLocale("#,###.0")  =>  12,345.8
7.1234f.toLocale("#.000")       =>  7.123
0.1234f.toLocale("#.000")       =>  .123
0.1234f.toLocale("0.00")        =>  0.12
70.12f.toLocale("0.0000")       =>  70.1200
```

Date, Time & DateTime toLocale

Couple simple DateTime examples:

```
DateTime.now.toLocale("kk:mmaa")              =>  09:10am
DateTime.now.toLocale("DDD 'of' MMMM, YYYY")  =>  9th of September, 2010
```

```
Str toLocale(Str? pattern := null)
```

Format this time according to the specified pattern. If pattern is null, then a localized default is used. Any ASCII letter in the pattern is interpreted as follows:

```
YY     Two digit year             07
YYYY   Four digit year            2007
M      One/two digit month        6, 11
MM     Two digit month            06, 11
MMM    Three letter abbr month    Jun, Nov
MMMM   Full month                 June, November
D      One/two digit day          5, 28
DD     Two digit day              05, 28
DDD    Day with suffix            1st, 2nd, 3rd, 24th
WWW    Three letter abbr weekday  Tue
```

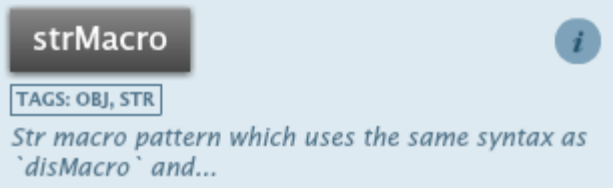
```
WWWW   Full weekday               Tuesday
V      One/two digit week of year 1,52
VV     Two digit week of year     01,52
VVV    Week of year with suffix   1st,52nd
h      One digit 24 hour (0-23)   3, 22
hh     Two digit 24 hour (0-23)   03, 22
k      One digit 12 hour (1-12)   3, 11
kk     Two digit 12 hour (1-12)   03, 11
m      One digit minutes (0-59)   4, 45
mm     Two digit minutes (0-59)   04, 45
s      One digit seconds (0-59)   4, 45
ss     Two digit seconds (0-59)   04, 45
SS     Optional seconds (only if non-zero)
f*     Fractional secs trailing zeros
F*     Fractional secs no trailing zeros
a      Lower case a/p for am/pm   a, p
aa     Lower case am/pm           am, pm
A      Upper case A/P for am/pm   A, P
AA     Upper case AM/PM           AM, PM
z      Time zone offset           Z, +03:00 (ISO 8601, XML Schema)
zzz    Time zone abbr             EST, EDT
zzzz   Time zone name             New_York
'xyz'  Literal characters
```

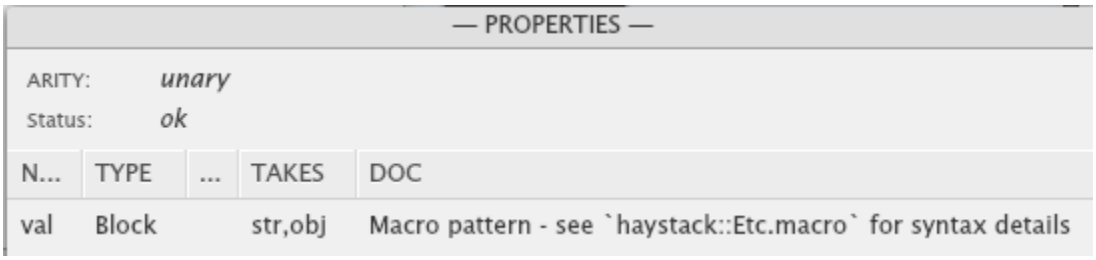
strLower

Changes all ASCI letters to lower case.

strMacro

Str macro pattern which uses the same syntax as `disMacro` and `haystack::Etc.macro`. The macro variables are resolves against the task's variables.





From http://licensing.intellastar.com/doc/haystack/Etc#macro

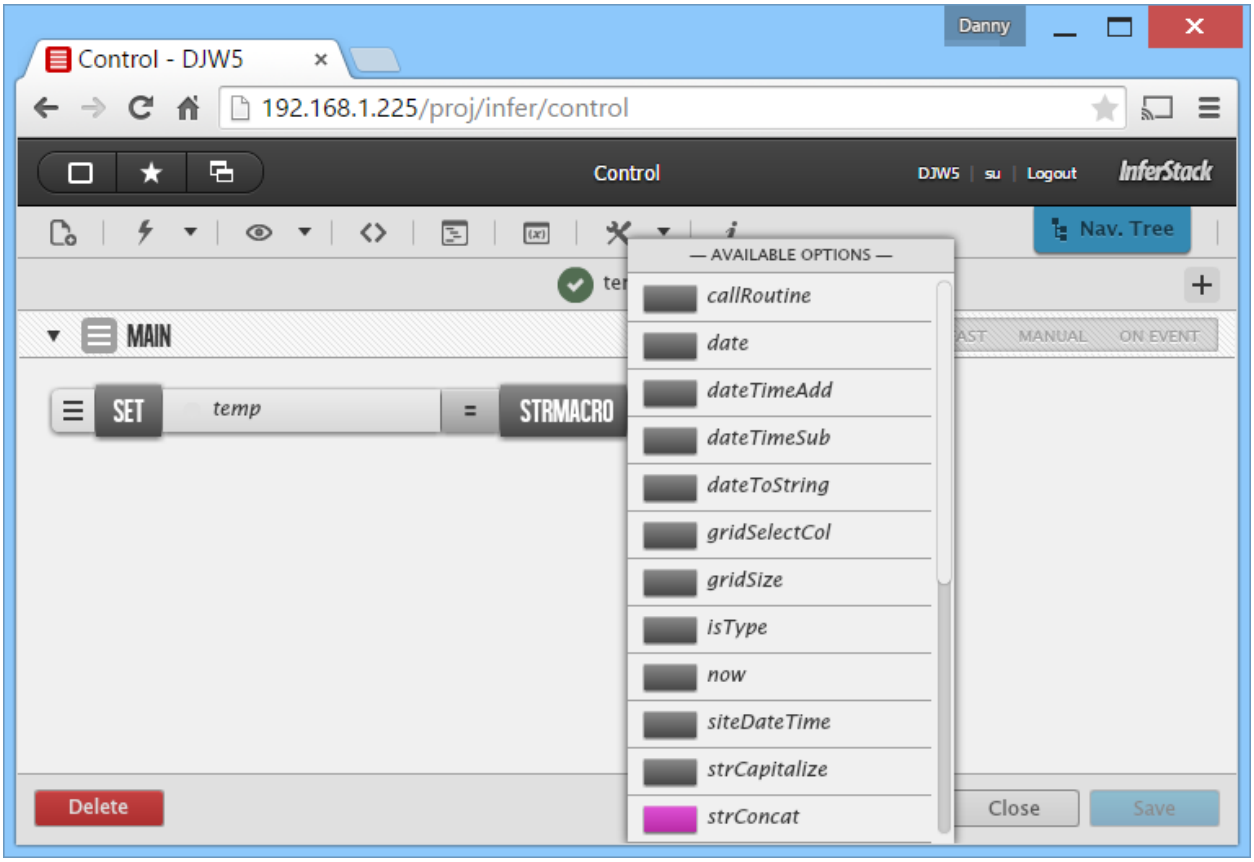
macro

```
const static Str macro(Str pattern, Dict scope)
```

Process macro pattern with given scope of variable name/value pairs. The pattern is a Unicode string with embedded expressions:

- `$tag`: resolve tag name from scope, variable name ends with first non-tag, see `Etc.isTagName`
- `${tag}`: resolve tag name from scope
- `$<pod::key>`: localization key

Any variables which cannot be resolved in the scope are returned as-is (such `$name`) in the result string. If a tag resolves to Ref, then we use Ref.dis for string.



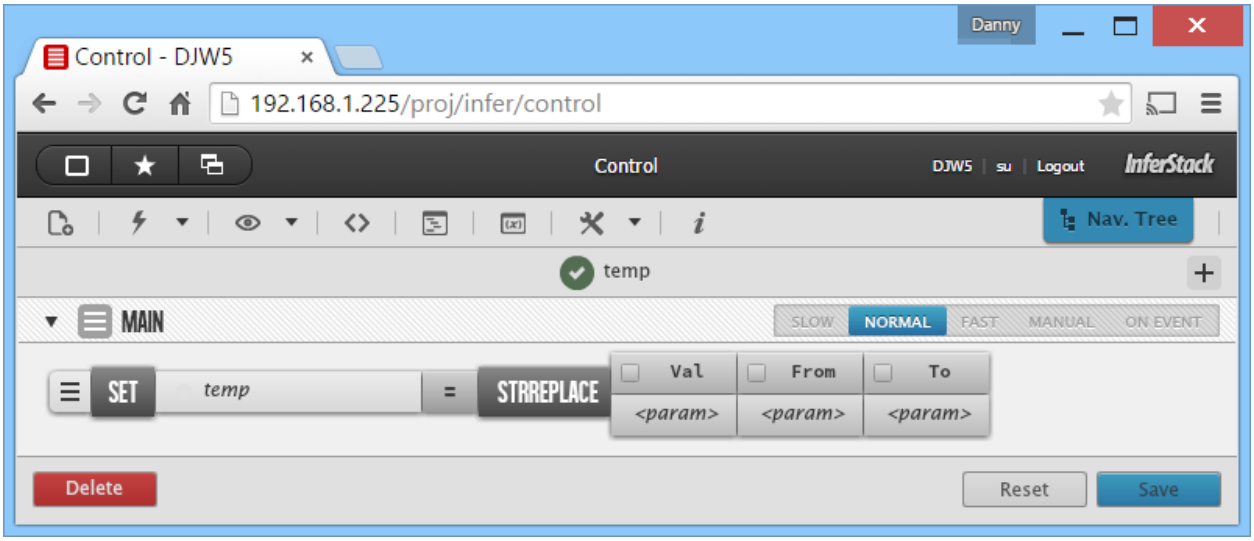
strReplace

String replace

— PROPERTIES —

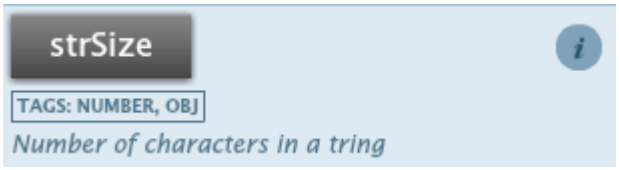
ARITY: --
Status: *ok*

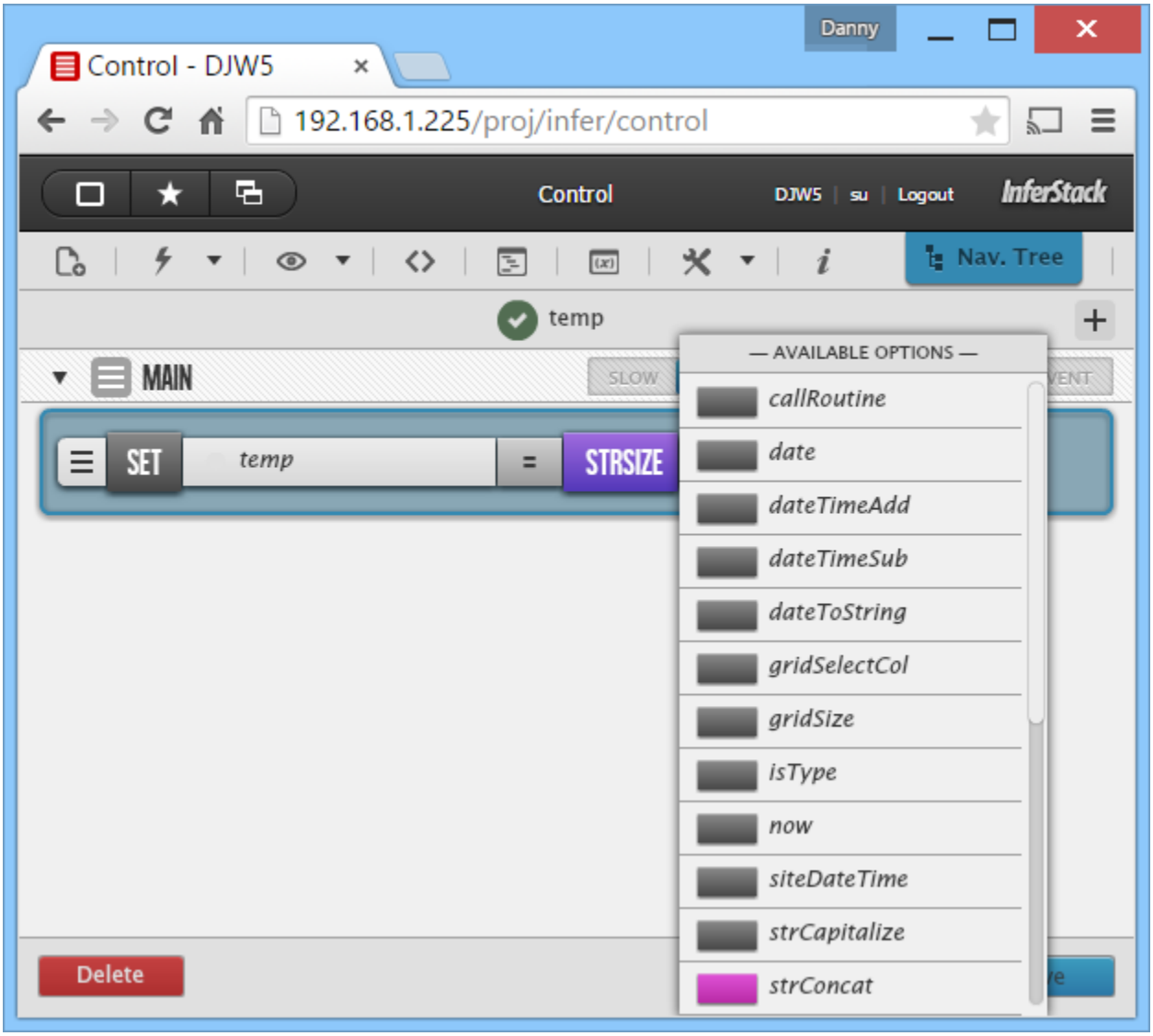
NAME	TYPE	Default	TAKES	DOC
val	Block		str,obj	Source string
from	Block		str,obj	Substring to find and re
to	Block		str,obj	Replacement substring



strSize

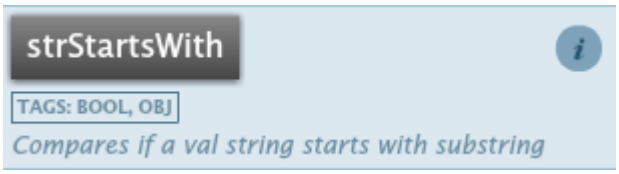
Number of characters in a string.





strStartsWith

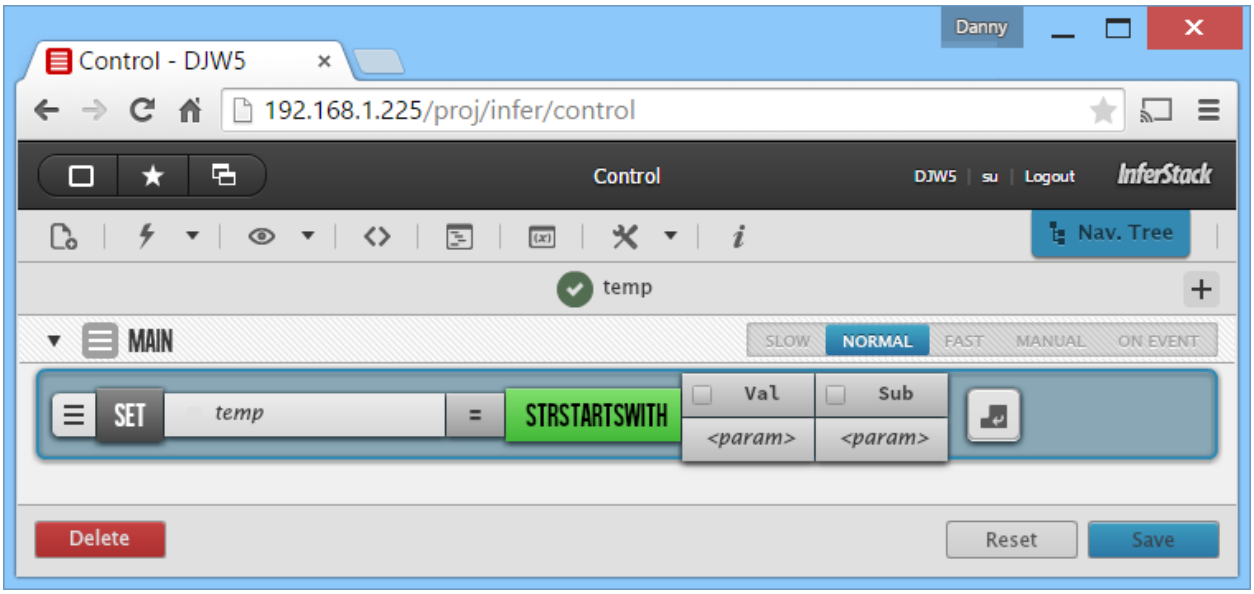
Compares if a val string starts with substring.



— PROPERTIES —

ARITY: --
Status: *ok*

NAME	TYPE	Default	TAKES	DOC
val	Block		str,obj	Source string
sub	Block		str,obj	Substring string to chec

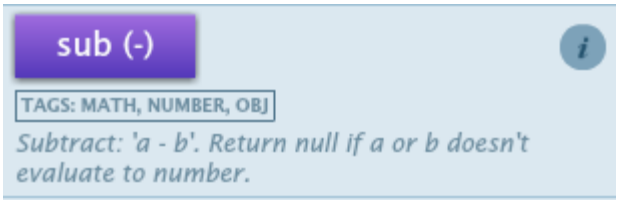


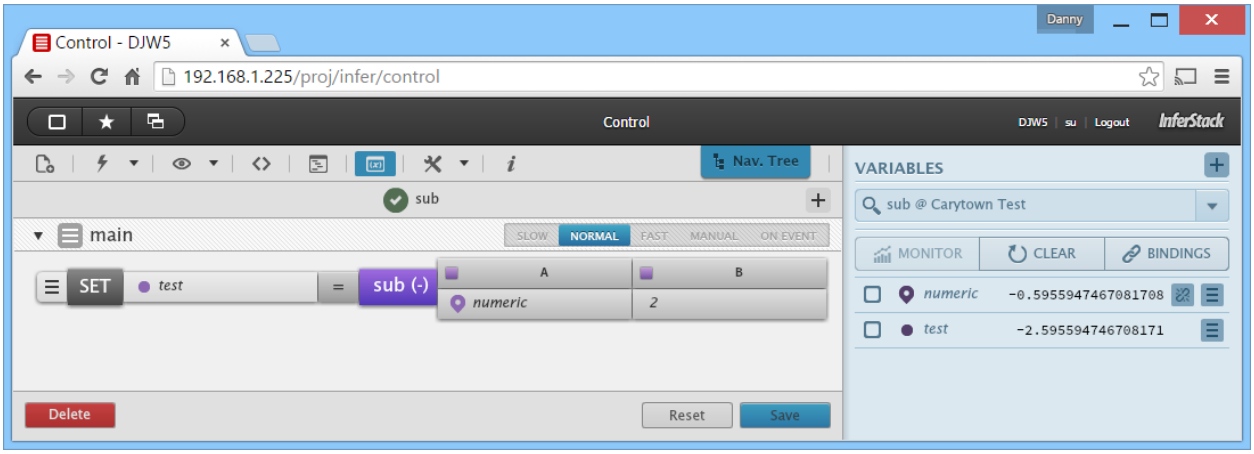
strUpper

Changes all ASCI letters to upper case.

sub

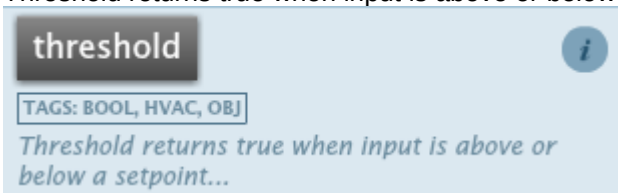
Subtract: 'a - b'. Return null if a or b doesn't evaluate to number.





threshold

Threshold returns true when input is above or below a setpoint with configurable deadband.

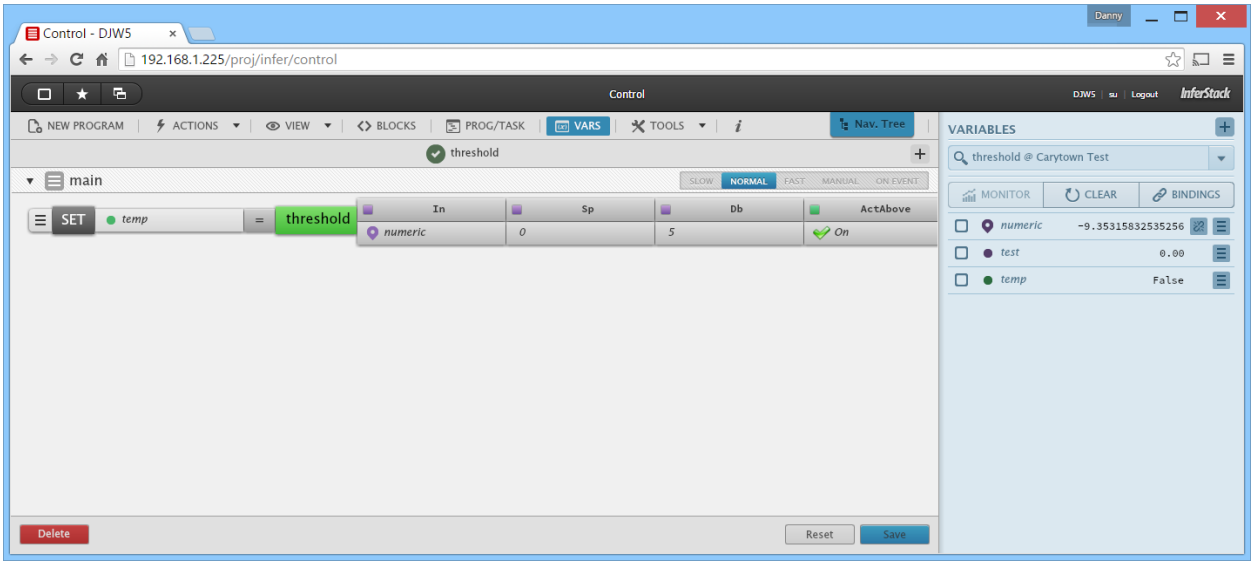


— PROPERTIES —

ARITY: --

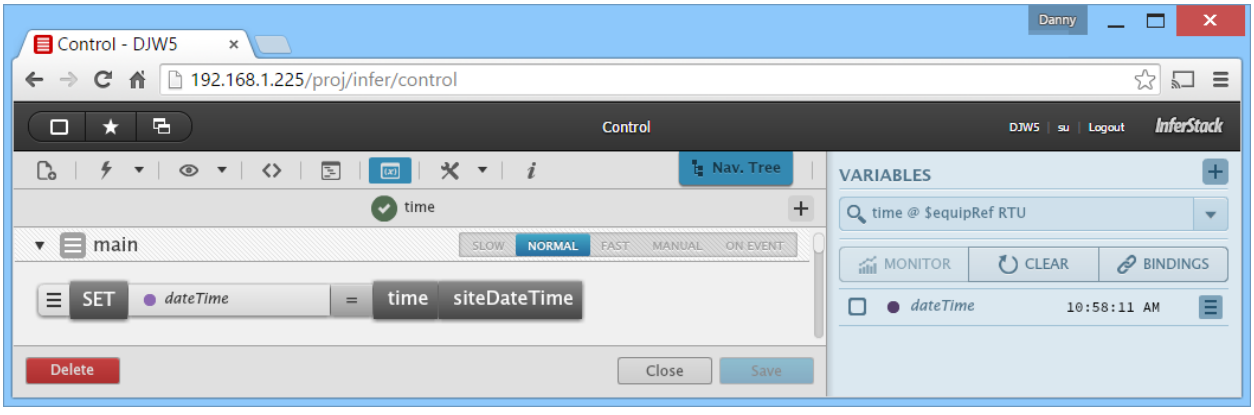
Status: *ok*

NAME	TYPE	Default	TAKES	DOC
in	Block		number,obj	Input variable
sp	Block		number,obj	Setpoint
db	Block		number,obj	Deadband value or if nu
actAbove	Block		bool,obj	Check if input is above



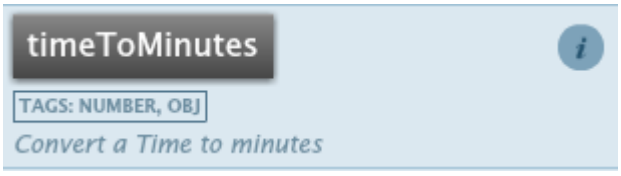
time

Get the Time part of a DateTime.



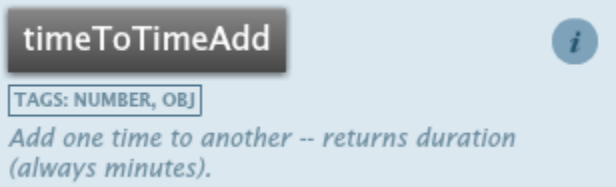
timeToMinutes

Convert a Time to minutes.



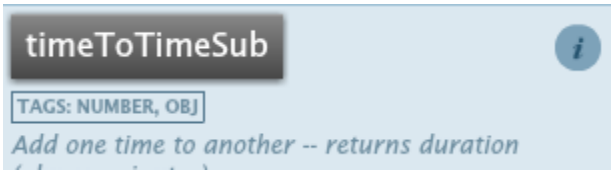
timeToTimeAdd

Add one time to another – returns duration (always minutes).



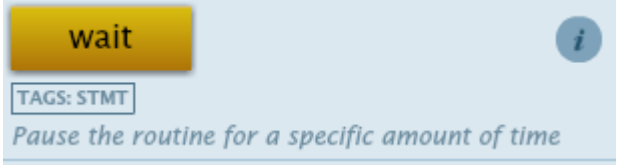
timeToTimeSub

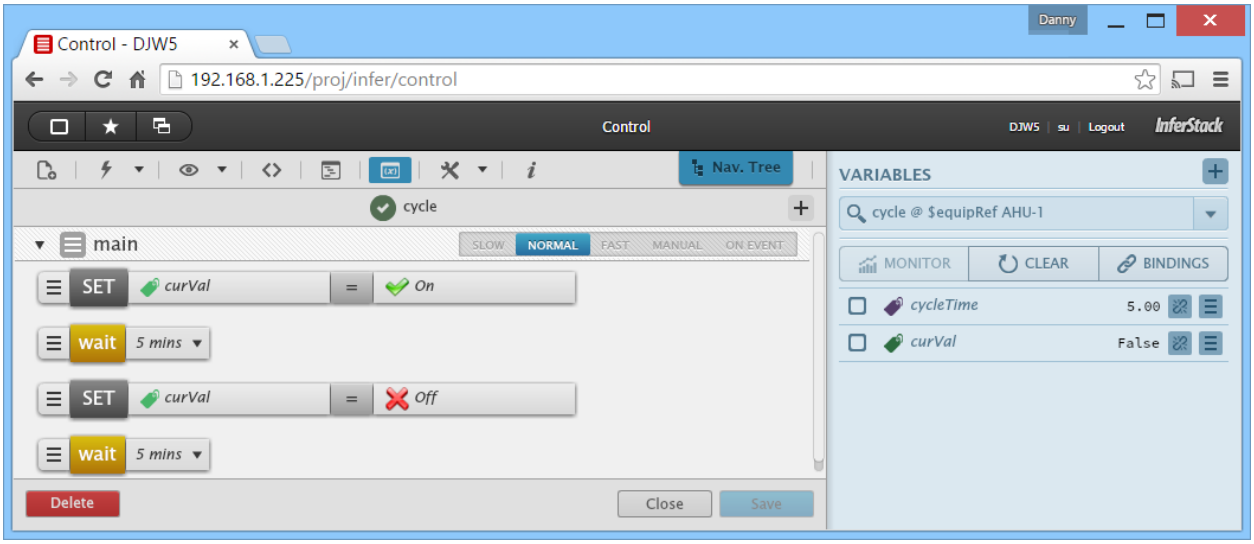
Subtract one time from another – returns duration (always minutes).



wait

Pause the routine for a specific amount of time.

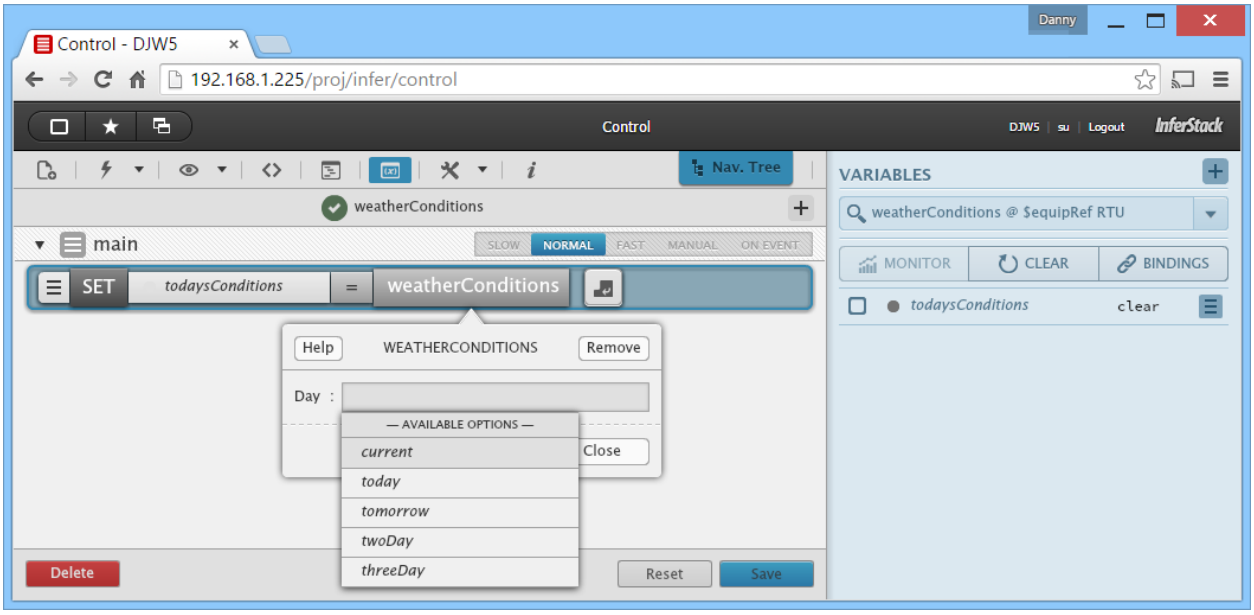


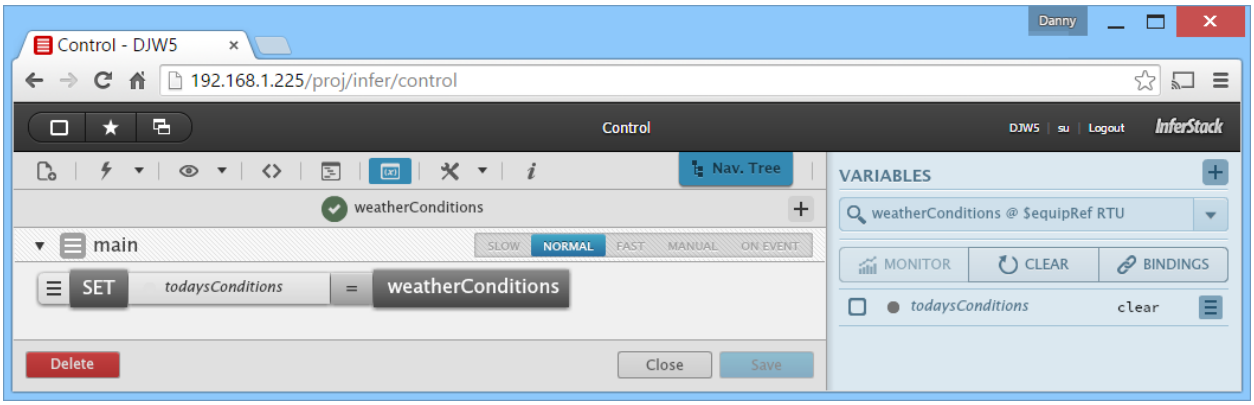


weatherConditions

Get the conditions from a forecast.

Note: You must have Weather defined on the site for the weather blocks to perform.

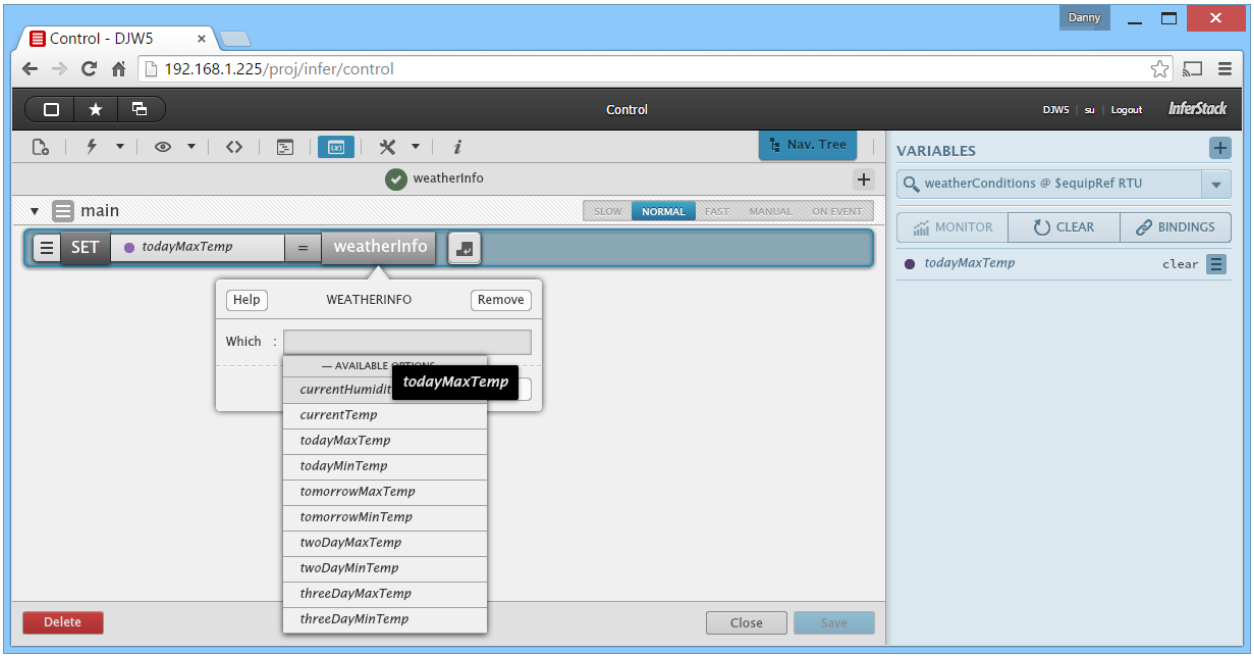




weatherInfo

Get the temperature information from a forecast.

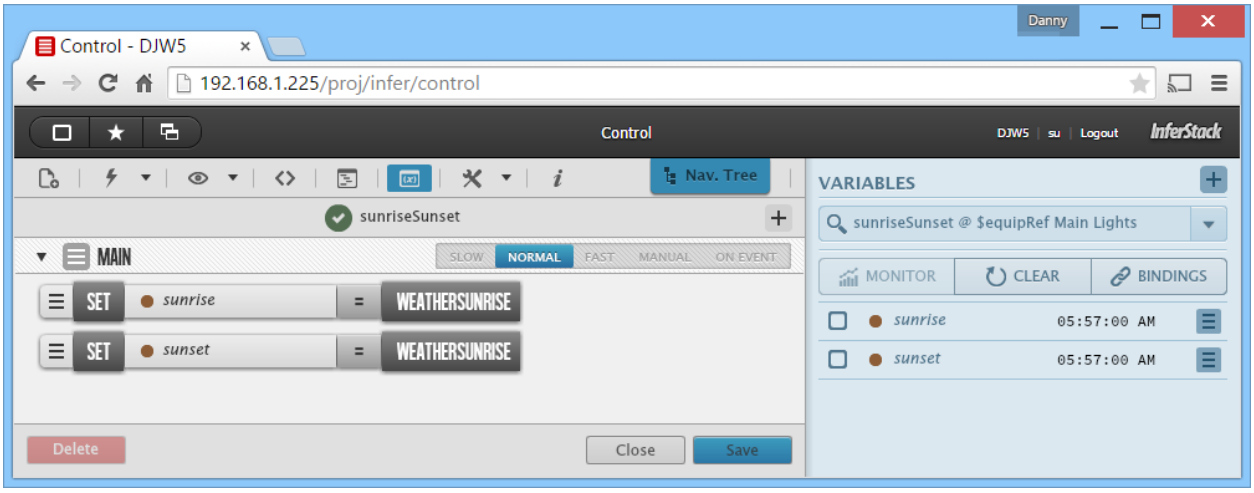
Note: You must have Weather defined on the site for the weather blocks to perform.



weatherSunrise

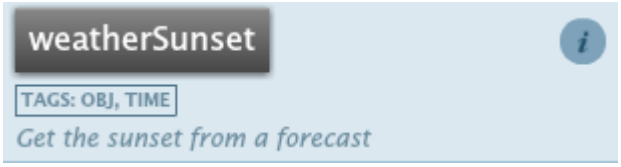
Get the sunrise from a forecast.

Note: You must have Weather defined on the site for the weather blocks to perform.

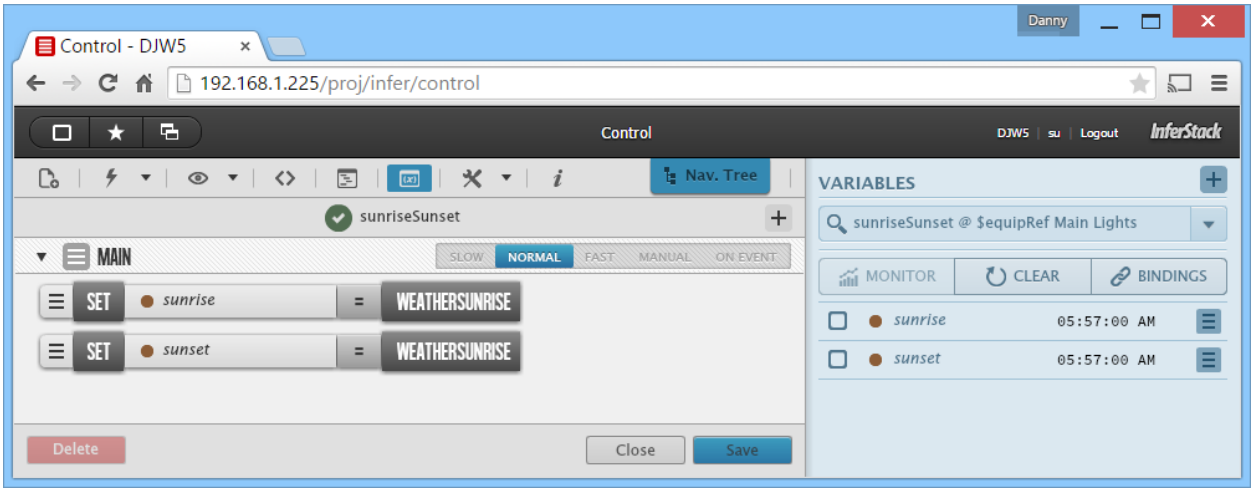


weatherSunset

Get the sunset from a forecast.

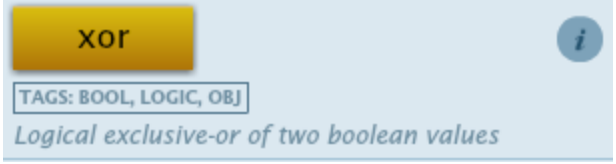


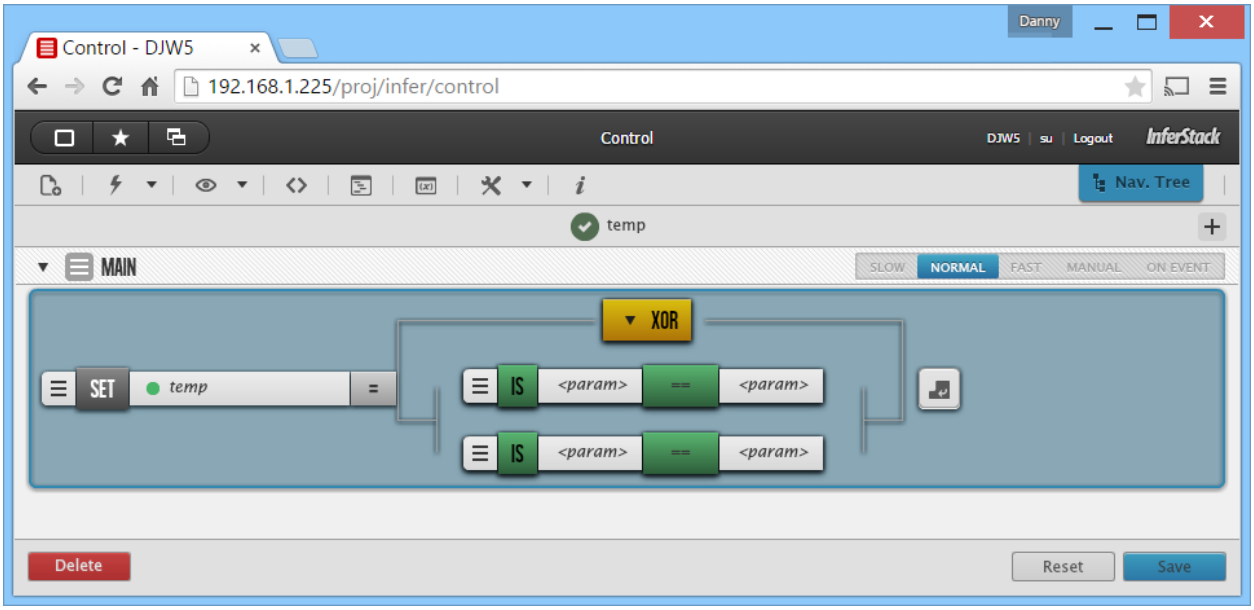
Note: You must have Weather defined on the site for the weather blocks to perform.



xor

Logical exclusive-or of two boolean values.

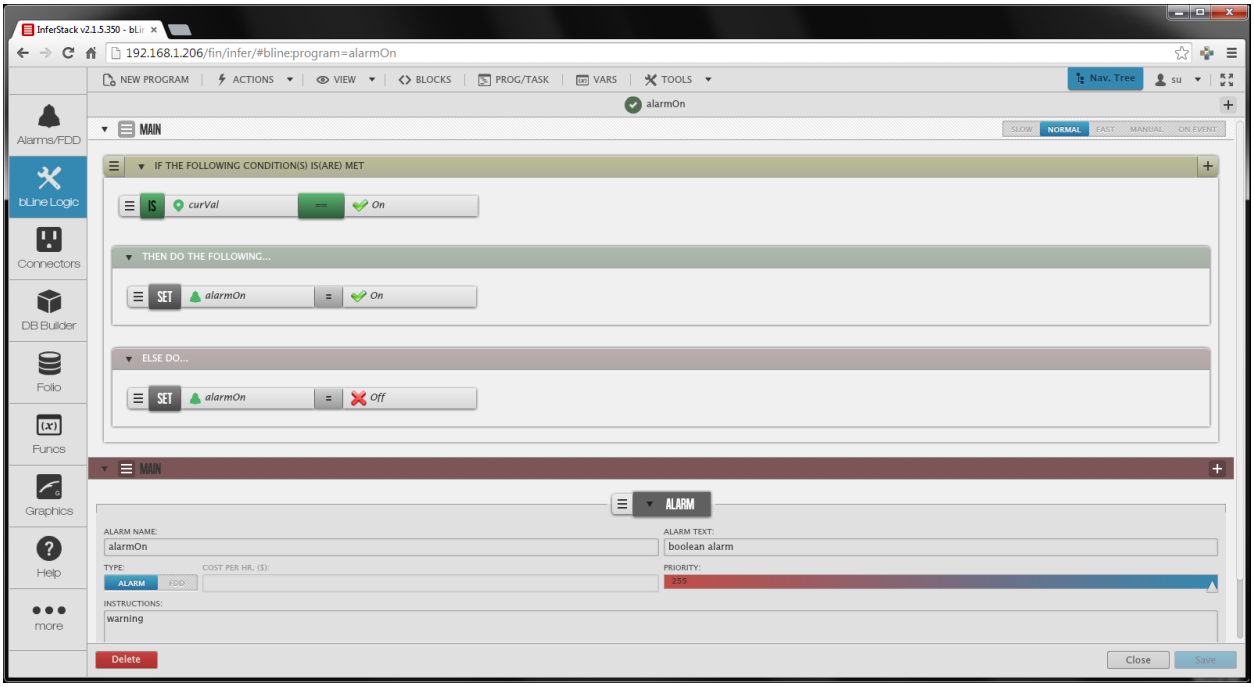




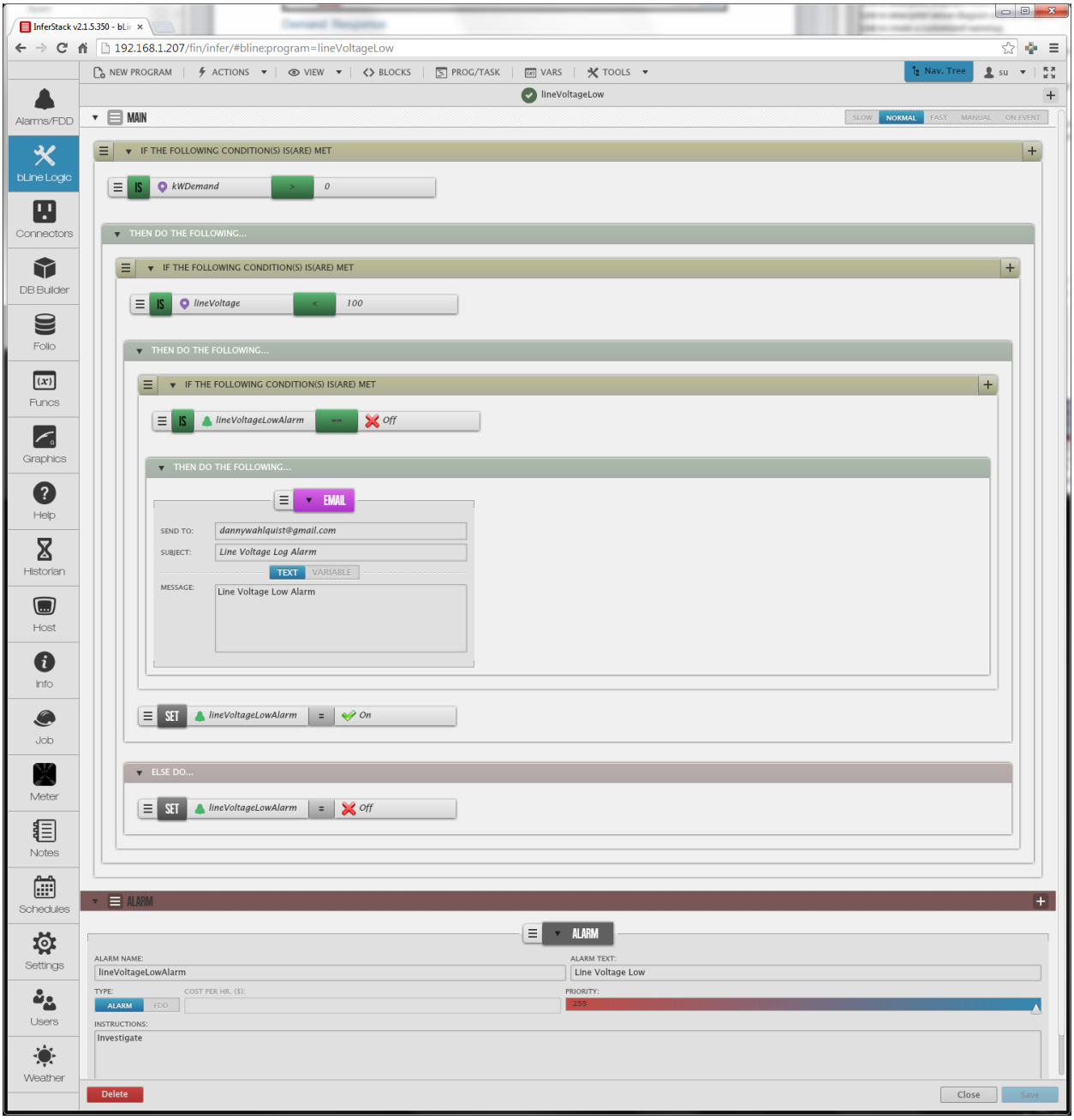
Examples

These examples may not perform the function you need, but are provided to help you see possible applications.

Alarm On

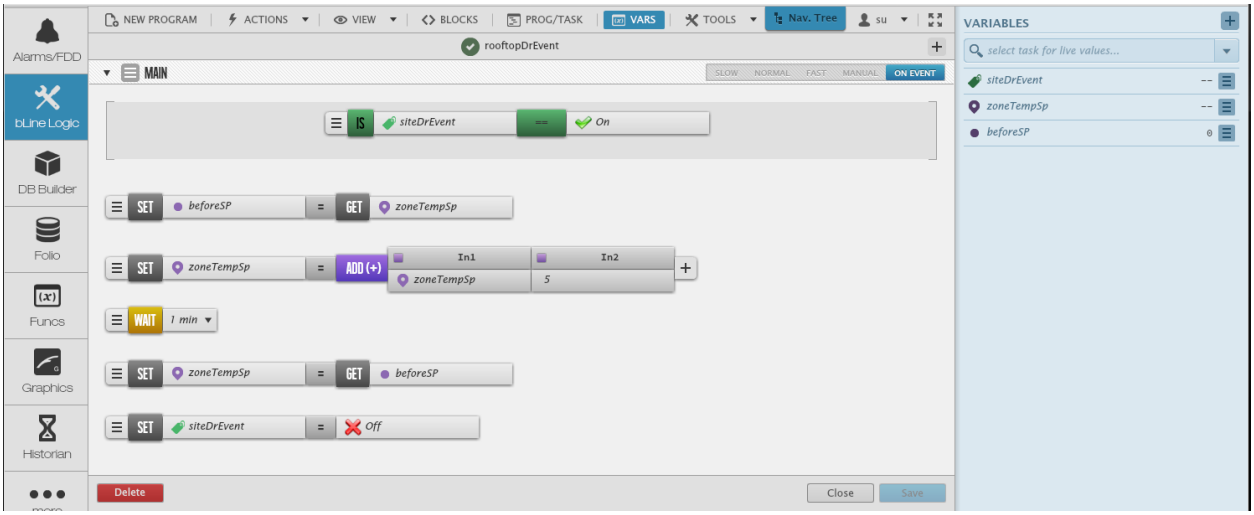


Demand Monitoring and Email Value



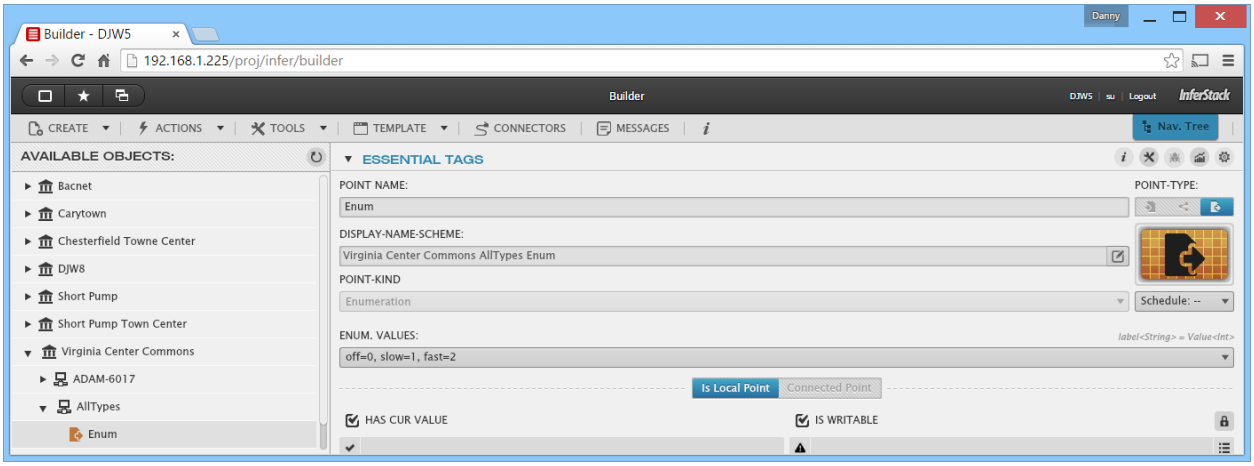
Note: The email above is only sent when alarm is first set.

Demand Response



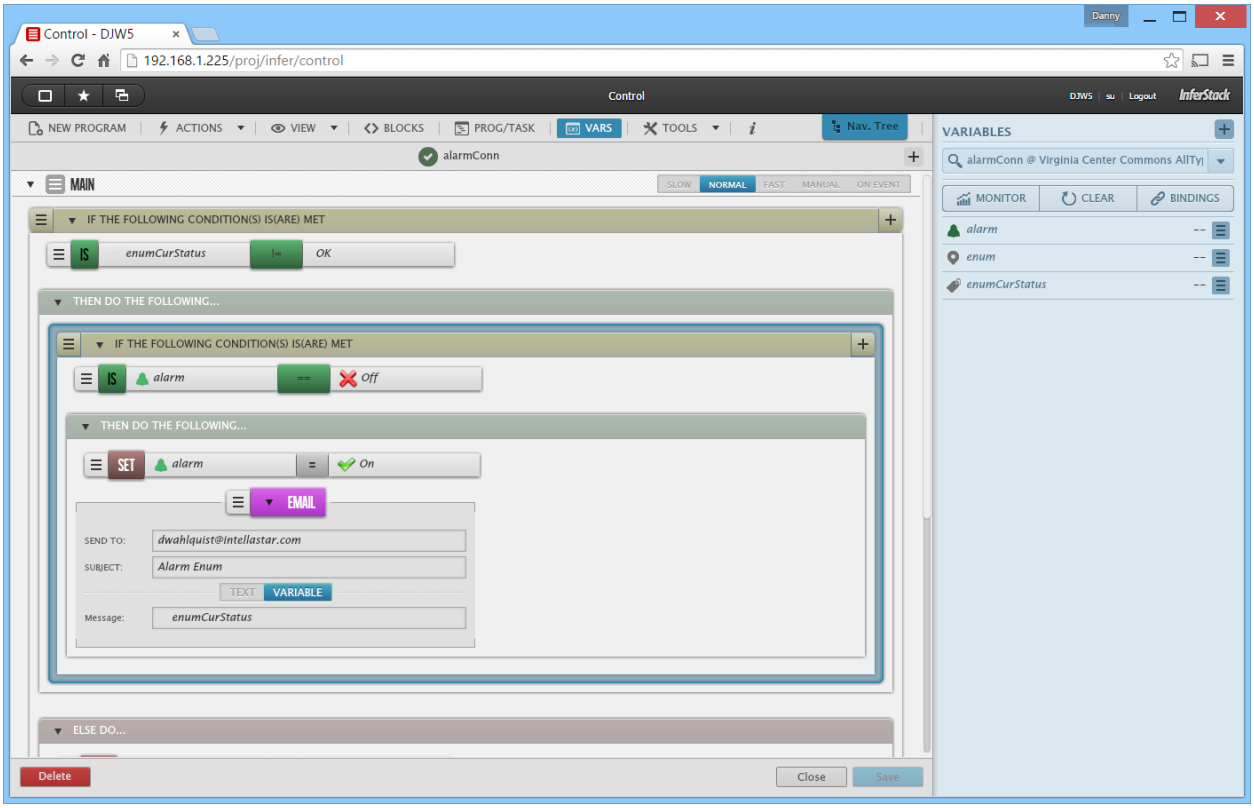
Enumeration

Set up an enumeration point with string values like this off=0, slow=1, fast=2 example.



Next write a control program to alarm. In this example I look for my enumCurStatus point value to not be OK, and then generate an alarm and email me if it was not already in alarm.

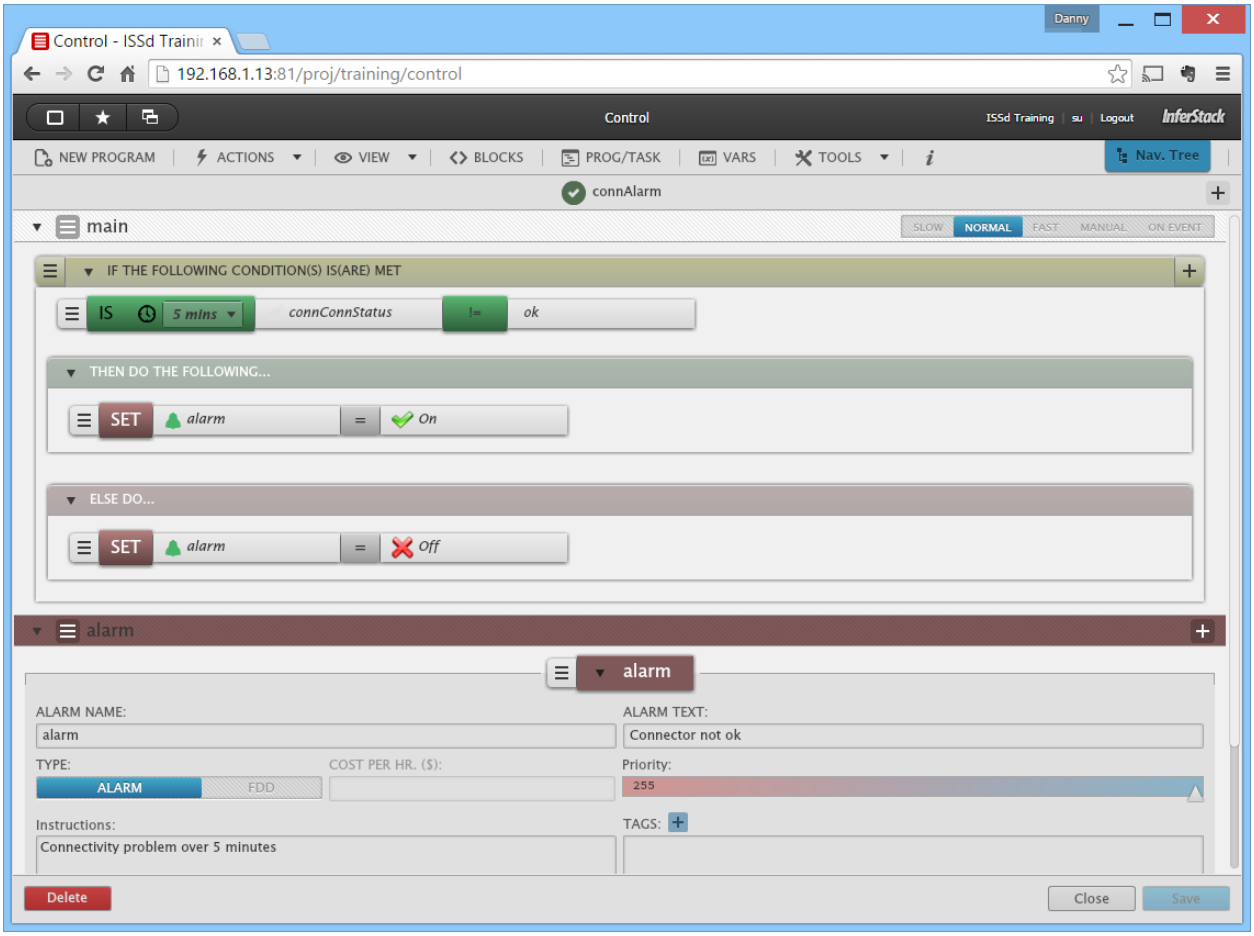
Note: Whenever entering a value in Control, always press the Enter key after typing your value.



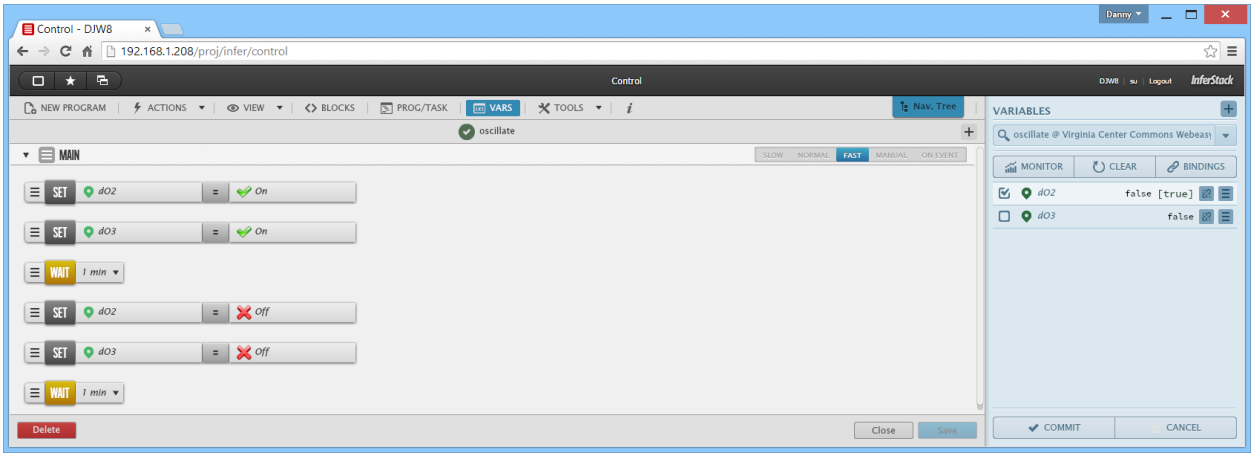
Note: The alarm email above is only sent when alarm is first set.

Monitor Connectors

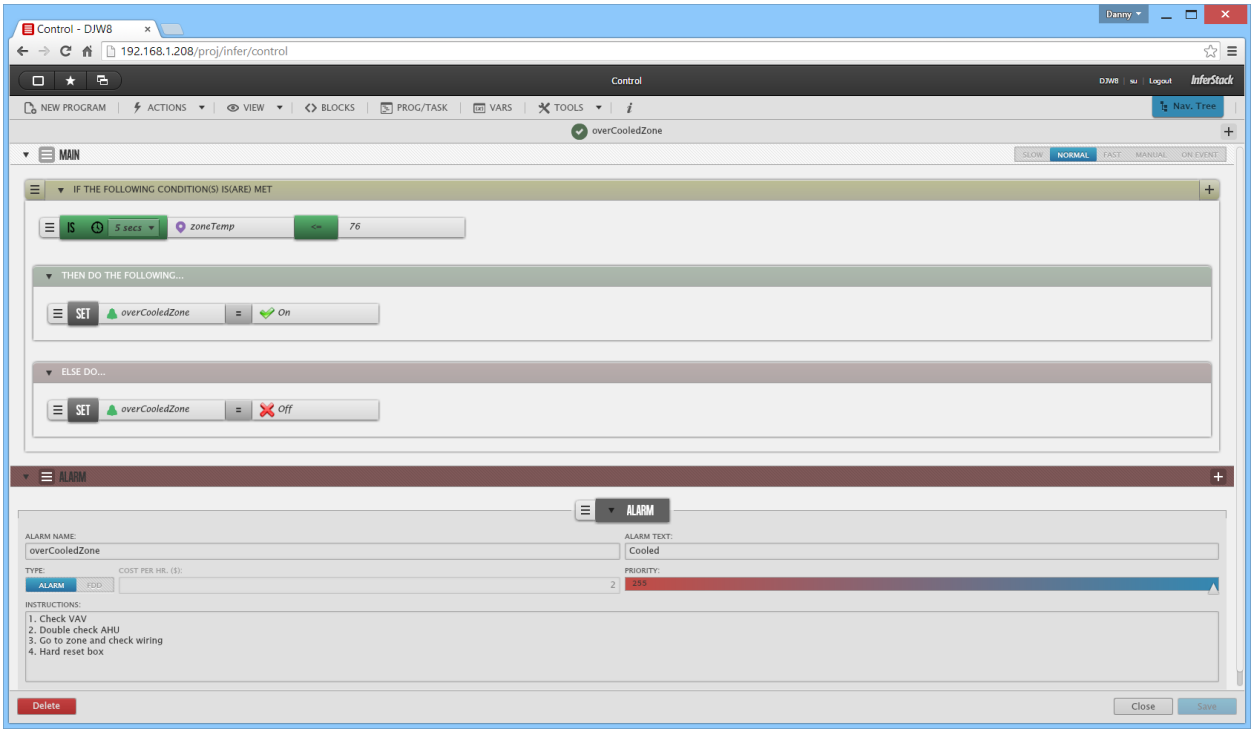
This program monitors connectors and alarms if there connStatus is not OK for 5 minutes.



Oscillate



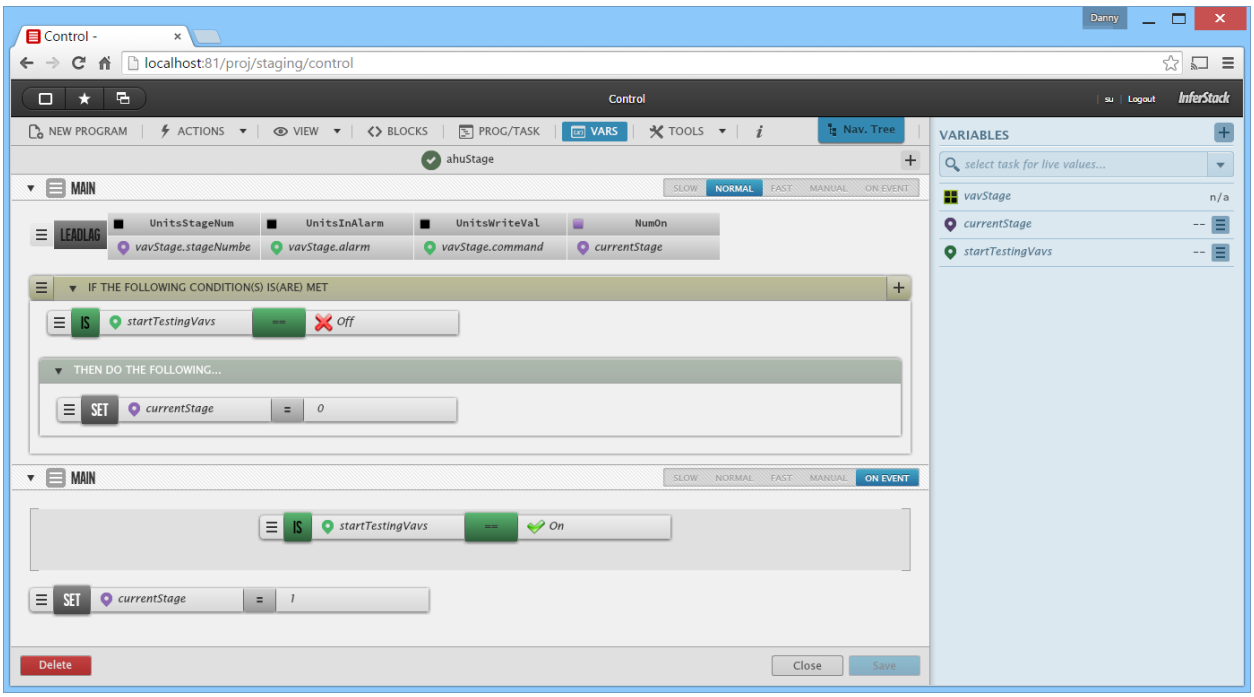
Over Cooled Zone



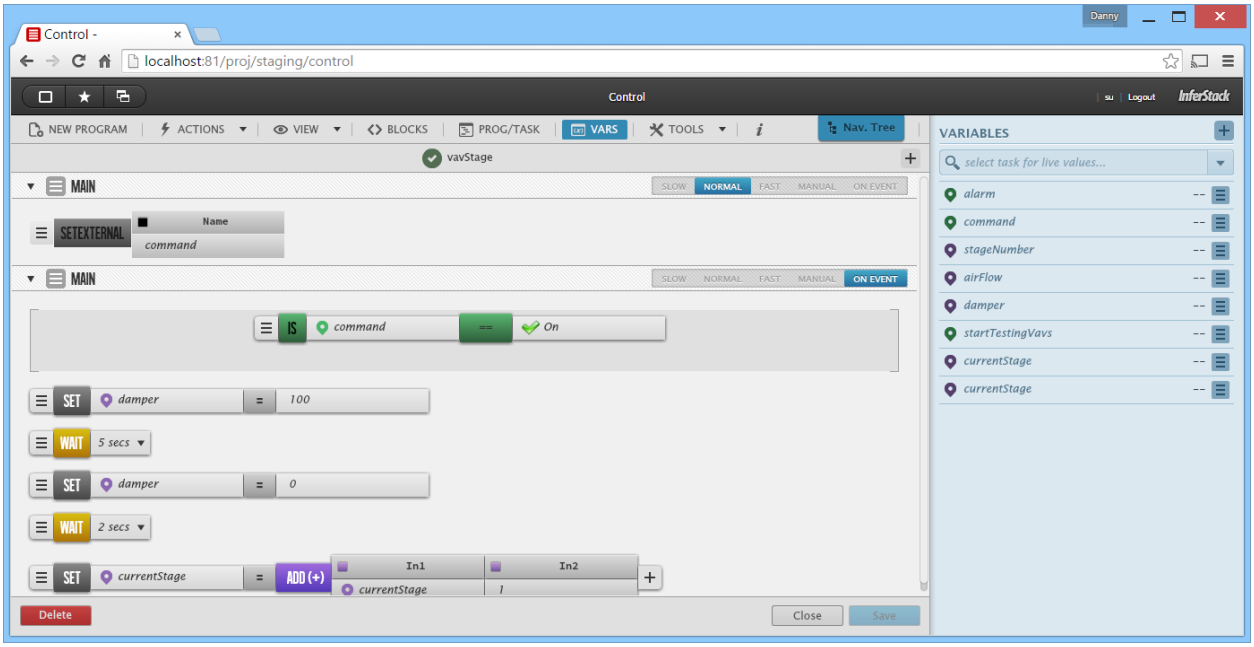
Staging

This example uses 2 programs to implement staging using the LeadLag and SetExternal blocks and On Event initiation.

ahuStage

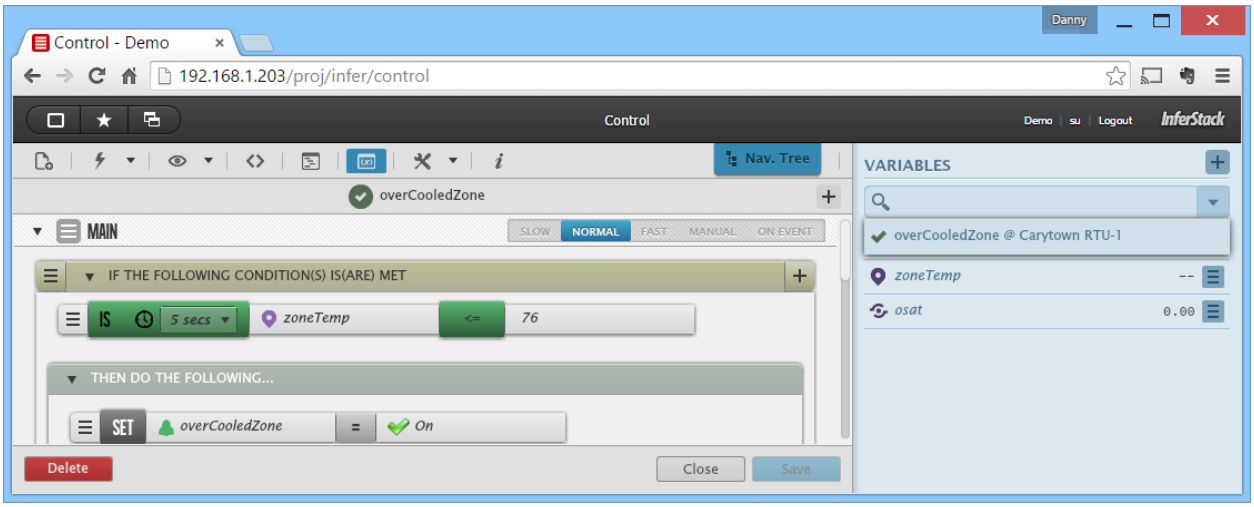


vavStage

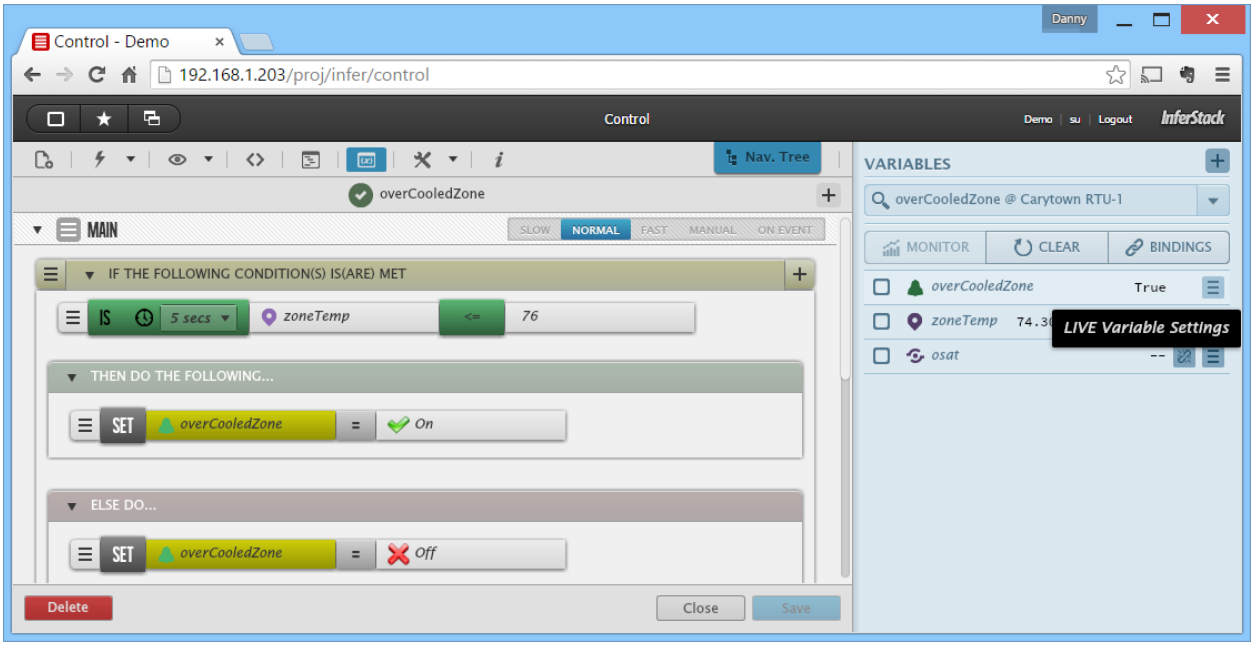


Troubleshooting

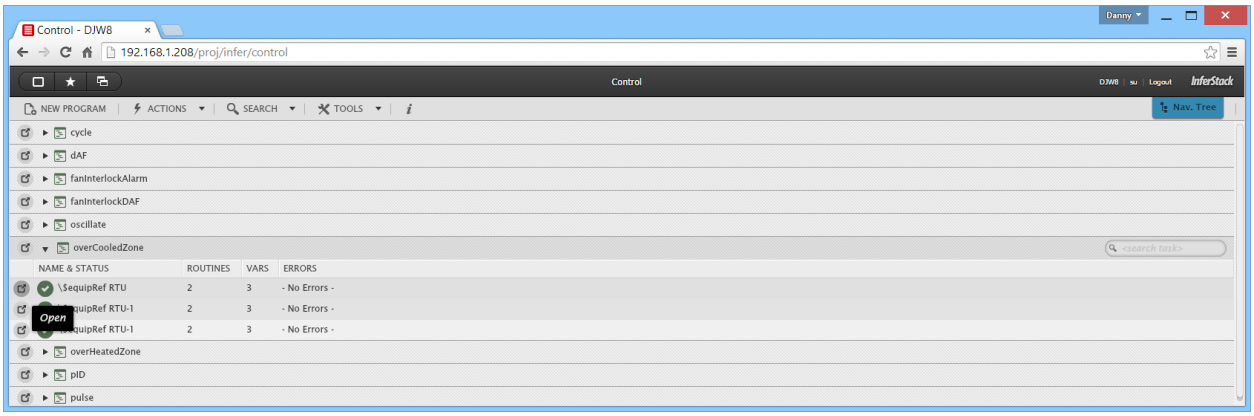
When trying to determine whether a Control program is performing properly, you will often want to put the program in debug mode and watch it variables. You do this by enabling Variables in the Menu and selecting the drop down.

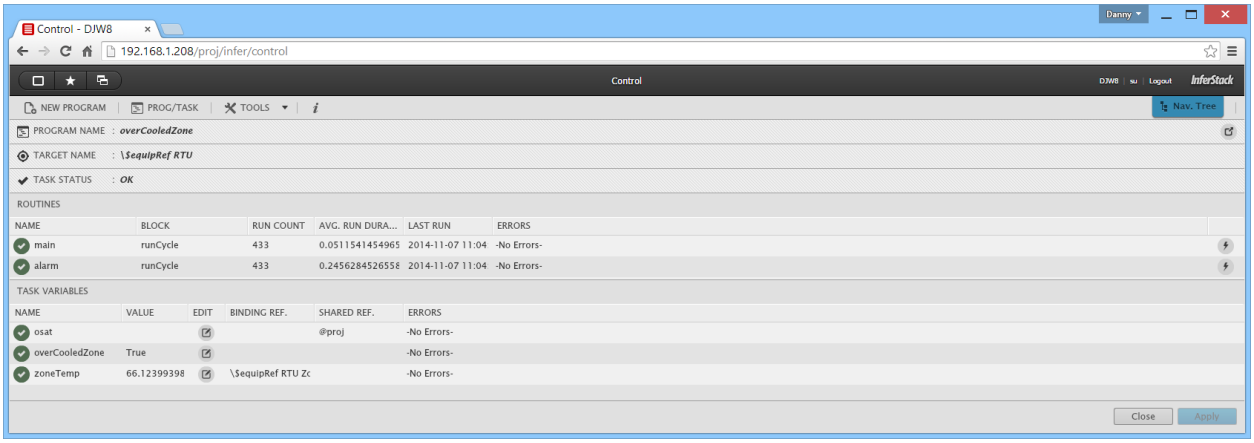


You now can see the variables update and change bindings to watch how the program responds.



Probably the most important piece of data is the runAvgDur reported on each routine. This will tell you how long it is taking to execute that routine (and the count will tell you how often). You get this by opening a routine.





The other key place to look is the Debug App - Threads which will show you what overall percentage of CPU time each thread is consuming (and how control engine thread compares to rest of system).

The Threads report is showing you two things:

1. Total time each thread has consumed of CPU time sorted by most CPU intensive thread down to lowest

2. The current state of each thread. An idle thread will be in the TIMED_WAITING mode

That screen will also show if you there is a deadlock (but that doesn't happen too often with the actor based concurrency):

http://fantom.org/doc/docLang/Actors.html

Reference

The reference information is on your device or server via the Help app or the portal at http://licensing.intellastar.com/ .