Kymera Utility Scripts Module

User Guide

1 800 470 2302
www.kymerasystems.com

Table of Contents

Introduction

The Kymera Utility Scripts module extends and simplifies utility functions within the Ignition Client and Designer by providing scripting and expression functions.

Usage

After installing the module, you will find a selection of scripting functions available in your script editor's autocomplete list as well as a selection of expression functions available in your Expressions Functions list. To see the list of scripting functions available to autocomplete, start by typing "system." In a script editor then press CTRL + SPACE. To see the list of expression functions, press the "Functions" icon from the expression editor.

Scripting Functions

The following scripting functions are available with the module.

system.alert.createNotifier

Description

Creates a notification dialog in the top right corner of the screen, on top of all other components. Multiple dialogs can be spawned and will stack in a column with the newest message on the bottom.

Syntax

```
system.alert.createNotifier(title, bodyText, time, bgColor, textColor)
```

Parameters

String `title` – The title to display in the notification dialog.
String `bodyText` – The message to appear in the dialog, below the title.
Integer `time` – The time, in milliseconds, that the dialog is visible.
Color `bgColor` – The background color of the dialog.
Color `textColor` – The text color of the title and body text.

Returns

nothing

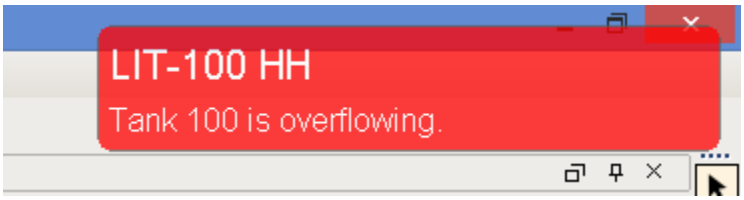
Scope

Client, Designer

Examples

The following snippet creates a dialog with a red background and white text that will display for 30 seconds.

```
system.alert.createNotifier('LIT-100 HH', 'Tank 100 is overflowing.', 30000, '#FF0000',
	'white')
```



system.client.moveProject

Description

Allows you to move a client window from one screen to another.

Syntax

```
system.client.moveProject()
```

Parameters

none

Returns

nothing

Scope

Client, Designer

```
system.client.moveProject(screenIndex)
```

Parameters

int `screenIndex` – The screen index to move the client window to.

Returns

nothing

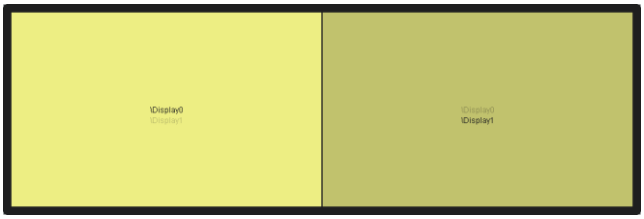
Scope

Client, Designer

Examples

This code snippet could be placed in a client Menubar script to allow users to move a full screen window from one screen to another. Once invoked, a popup displaying available screens allows you to select which screen to move window to.

```
system.client.moveProject()
```



This code snippet would move the client window to the second monitor.

```
system.client.moveProject(1)
```

system.ctg.addPen

Description

Adds a pen to the Kymera CTG graph.

Syntax

```
system.ctg.addPen(tagpath)
```

Parameters

String `tagpath` – The tag path to the tag you want to add as a pen.

Returns

nothing

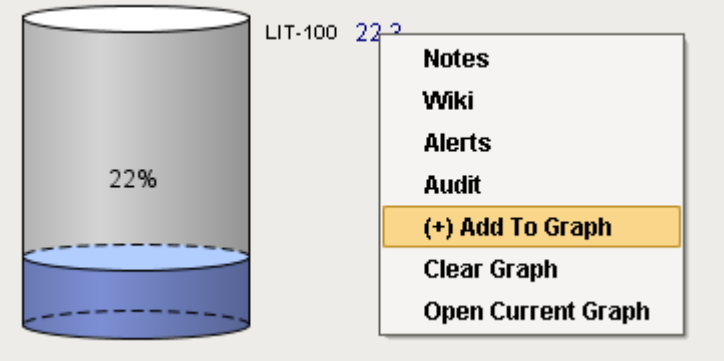
Scope

Client, Designer

Examples

You could build a pop-up menu on an analog template with a tagPath property and add the tag to the Kymera CTG graph.

```
system.ctg.addPen('analog/lit-100')
```



system.ctg.clearPens

Description

Clears all pens from the Kymera CTG graph.

Syntax

```
system.ctg.clearPens()
```

Parameters

none

Returns

nothing

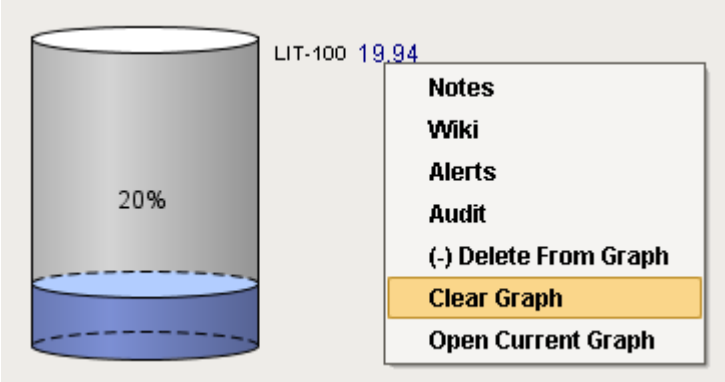
Scope

Client, Designer

Examples

You could build a pop-up menu on an analog template to remove all tags from the Kymera CTG graph.

```
system.ctg.clearPens()
```



system.ctg.deletePen

Description

Deletes the specified pen from the Kymera CTG graph.

Syntax

```
system.ctg.deletePen(tagpath)
```

Parameters

String `tagpath` - The tag path to the pen you want to delete.

Returns

nothing

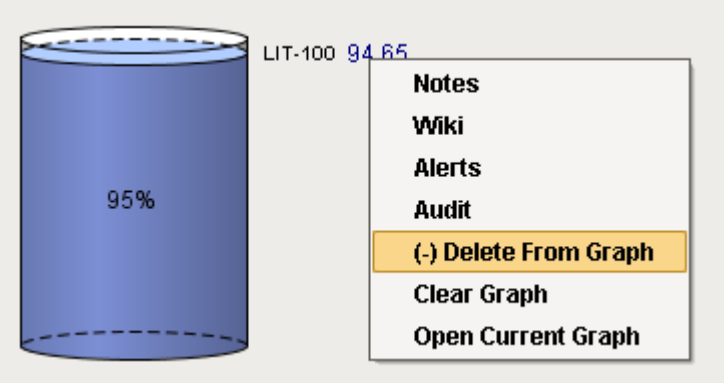
Scope

Client, Designer

Examples

You could build a pop-up menu on an analog template with a tagPath property and remove the tag from the Kymera CTG graph.

```
system.ctg.deletePen('analog/lit-100')
```



system.ctg.findPen

Description

Checks if the given pen exists on the Kymera CTG graph.

Syntax

```
system.ctg.findPen(tagpath)
```

Parameters

String `tagpath` - The tag path of the pen you want to find.

Returns

int – The index of the pen in the internal list of pens on the Kymera CTG graph, or -1 if the pen does not exist.

Scope

Client, Designer

Examples

This snippet would return the index of the analog tag LIT-100.

```
system.ctg.findPen('analog/lit-100')
```

system.ctg.getDatabaseConnections

Description

Gets a list of active database connections.

Syntax

```
system.ctg.getDatabaseConnections()
```

Parameters

none

Returns

PyList – A PyList containing the active database connections.

Scope

Client, Designer

Examples

```
datasources = system.ctg.getDatabaseConnections()
```

system.ctg.getPensList

Description

Gets a list of the pens on the Kymera CTG graph.

Syntax

```
system.ctg.getPensList()
```

Parameters

none

Returns

PyList – A PyList containing the pens on the Kymera CTG graph.

Scope

Client, Designer

Examples

```
pens = system.ctg.getPensList()
```

system.ctg.getPensString

Description

Gets a comma separated string containing the pens on the Kymera CTG graph.

Syntax

```
system.ctg.getPensString()
```

Parameters

none

Returns

String – A comma separated string of the pens on the Kymera CTG graph.

Scope

Client, Designer

Examples

```
pens = system.ctg.getPensString()
```

system.ctg.install

Description

Adds the Kymera CTG graph database tables to the specified datasource and imports the CTG windows into the current project.

The following tables are added:

ctg_axes, ctg_pens, ctg_saved_graph_pens, ctg_saved_graphs, ctg_subplots

The following windows are imported into a folder named 'CTG':

Add_pens, Axis_Edit, Pens_Axes, Subplot_Edit, Graph, Bulk_Pen_Creation, Pen_Edit, Graph_Save

Syntax

```
system.ctg.install(datasource)
```

Parameters

String `datasource` - The datasource to add the CTG database tables to.

Returns

nothing

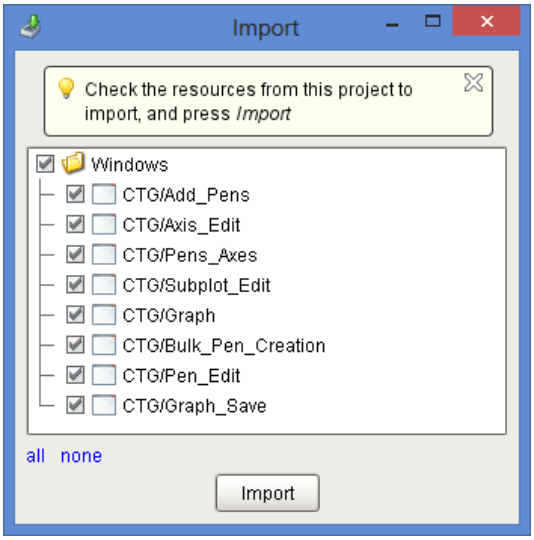
Scope

Client, Designer

Examples

This code snippet will install the CTG database tables to the default datasource and import the CTG windows into the current project.

```
system.ctg.install('')
```



system.ctg.openGraph

Description

Opens the Kymera CTG graph window.

Syntax

```
system.ctg.openGraph()
```

Parameters

none

Returns

nothing

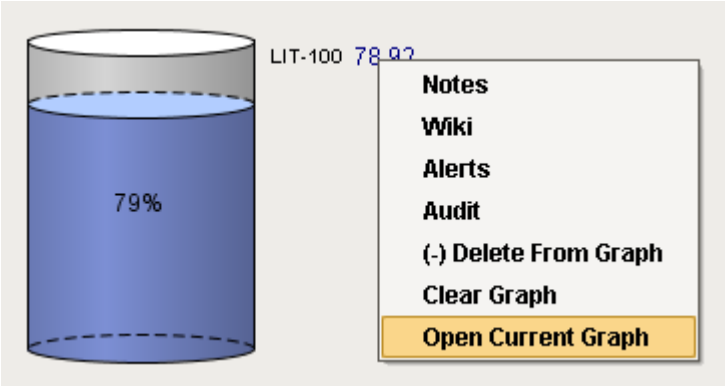
Scope

Client, Designer

Examples

You could build a pop-up menu on an analog template and open the Kymera CTG graph.

```
system.ctg.openGraph()
```



system.ctg.updateGraph

Description

Updates the Kymera CTG graph.

Syntax

```
system.ctg.updateGraph()
```

Parameters

none

Returns

nothing

Scope

Client, Designer

Examples

Use this method after adding or removing pens from the Kymera CTG graph to update the graph with the new pens.

```
system.ctg.updateGraph()
```

system.menu.addText

Description

Adds text to the menu bar in the Ignition client window. Subsequent text added will be added to the right side of existing text on the menu bar.

Syntax

```
system.menu.addText(text)
```

Parameters

String `text` – The text to display on the menu bar.

Returns

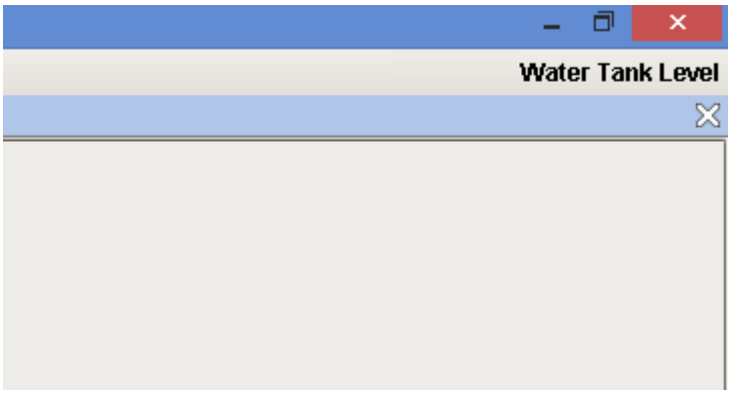
nothing

Scope

Client, Designer

Examples

```
system.menu.addText('Water Tank Level')
```



system.menu.addTag

Description

Adds tag data to the menu bar in the Ignition client window. The displayed data will update as the tag data updates.

Syntax

```
system.menu.addTag(tagPath, format)
```

Parameters

String `tagpath` – The tagpath to the tag you want to display.

String `format` – The format string for numerical data to display.

Returns

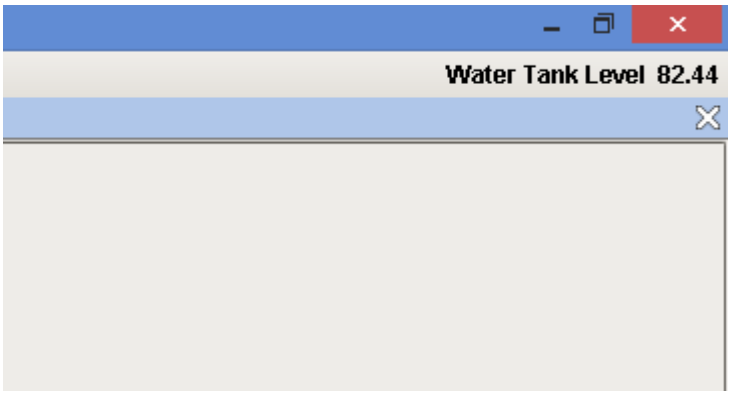
nothing

Scope

Client, Designer

Examples

```
system.menu.addTag('analog/lit100/eu', '#,##0.##')
```



system.menu.remove

Description

Removes specified item from the menu bar. Text added using addText() is added as a whole, so it must be removed as a whole.

Syntax

```
system.menu.remove(string)
```

Parameters

String `string` – The string to remove from the menu bar, or the tagpath of the item to remove from the menu bar.

Returns

nothing

Scope

Client, Designer

Examples

This snippet removes text added to the menu bar with addText().

```
system.menu.remove('Preparing Export Data…')
```

system.menu.remove('Preparing') will not remove just the word 'Preparing', since the full text must be matched.

This snippet removes tag data dded to the menu bar with addTag().

```
system.menu.remove('analog/lit100/eu')
```

system.menu.clearMenu

Description

Removes all added items from the menu bar.

Syntax

```
system.menu.clearMenu()
```

Parameters

none

Returns

nothing

Scope

Client, Designer

Examples

```
system.menu.clearMenu()
```

system.net.httpGetBytes

Description

Returns data from a URL as a byte array.

Syntax

```
system.net.httpGetBytes(urlString)
```

Parameters

String `urlString` – The url to get data from.

Returns

byte[] – A byte array containing the data from the URL.

Scope

All

Examples

This snippet would grab a PDF file from the web, save it to the Temp folder and open the PDF.

```
bytes = system.net.httpGetBytes('http://files.inductiveautomation.com/sellsheets/
Intro_to_Ignition.pdf ')
file = system.file.getTempFile('pdf')
system.file.writeFile(file, bytes)
system.net.openURL(file)
```

system.stats.calculateMean

Description

Calculates the mean of the data sample.

Syntax

```
system.stats.calculateMean(values, includeNullAndNaN)
```

Parameters

Number[] `values` – An array of numeric type values to calculate the mean of.

Boolean `includeNullAndNan` – If false (0), Null and NaN values will be ignored. If True (1), any Null and NaN values present will cause a return value of NaN.

Returns

Double – The mean of the data sample.

Scope

All

Examples

```
system.stats.calculateMean([1, 2, 3, None], 0)
```

…returns 2.0

```
system.stats.calculateMean([1, 2, 3, None], 1)
```

...returns NaN

system.stats.calculateMedian

Description

Calculates the median of the data sample. The data will sorted automatically in ascending order.

Syntax

```
system.stats.calculateMedian(values)
```

Parameters

Number[] `values` – An array of numeric type values to calculate the median of. Null values are not permitted.

Returns

Double – The median of the data sample.

Scope

All

Examples

```
system.stats.calculateMedian([1, 2, 5, 4, 3])
```

…returns 3.0

system.stats.getStdDev

Description

Calculates the standard deviation of the data sample.

Syntax

```
system.stats.getStdDev(data)
```

Parameters

Number[] `data` - An array of numeric type values to calculate the standard deviation of. Nulls are not permitted.

Returns

Double – The standard deviation of the data sample.

Scope

All

Examples

```
system.stats.getStdDev([2,4,4,4,5,5,7,9])
```

…returns 2.1380899353

system.stats.getLinearFit

Description

Fits a straight line to a set of (x, y) data, returning the slope and intercept.

Syntax

```
system.stats.getLinearFit(xData, yData)
```

Parameters

Number[] `xData` – An array containing the x-data. Nulls are not permitted.

Number[] `yData` - An array containing the y-data. Nulls are not permitted.

Returns

Double[] – An array with the intercept in [0] and the slope in [1].

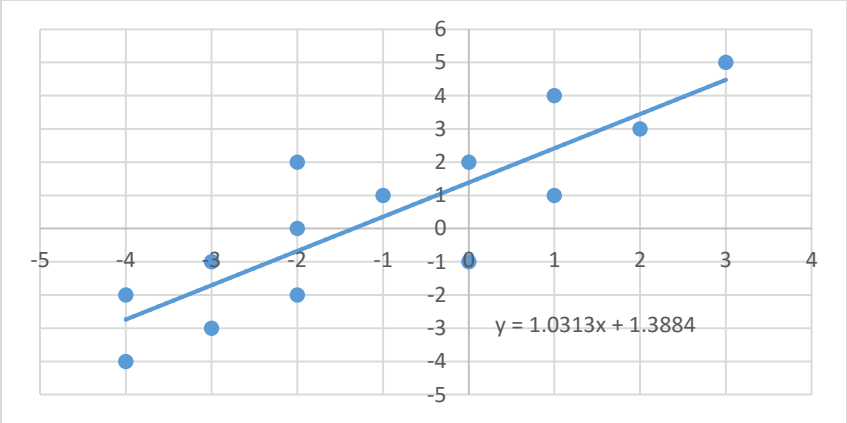
Scope

All

Examples

```
xData = [-4, -4, -3, -3, -2, -2, -2, -1, 0, 0, 1, 1, 2, 3]
yData = [-4, -2, -3, -1, 2, 0, -2, 1, 2, -1, 4, 1, 3, 5]
linearFit = system.stats.getLinearFit(xData, yData)
print 'Intercept: ' + str(linearFit[0]), ', Slope: '+ str(linearFit[1])
```

…returns Intercept: 1.38839285714 , Slope: 1.03125



Data plotted to show regression line, intercept and line equation.

system.stats.getSlope

Description

Finds the slope of a regression line using least squares.

Syntax

```
system.stats.getSlope(xData, yData)
```

Parameters

Number[] `xData` – An array containing the x-data. Nulls are not permitted.

Number[] `yData` - An array containing the y-data. Nulls are not permitted.

Returns

double – The slope of the regression line.

Scope

All

Examples

```
xData = [-4, -4, -3, -3, -2, -2, -2, -1, 0, 0, 1, 1, 2, 3]
yData = [-4, -2, -3, -1, 2, 0, -2, 1, 2, -1, 4, 1, 3, 5]
print system.stats.getSlope(xData, yData)
```

…returns 1.03125

system.stats.getCorrelation

Description

Calculates the correlation between two datasets. Both arrays should contain the same number of items. Null values are treated as zero.

Syntax

```
system.stats.getCorrelation(data1, data2)
```

Parameters

Number[] `data1` – The first dataset.

Number[] `data2` – The second dataset.

Returns

double – The correlation between the datasets.

Scope

All

Examples

```
xData = [-4, -4, -3, -3, -2, -2, -2, -1, 0, 0, 1, 1, 2, 3]
yData = [-4, -2, -3, -1, 2, 0, -2, 1, 2, -1, 4, 1, 3, 5]
print system.stats.getCorrelation(xData, yData)
```

…returns 0.853501266116

system.stats.getMovingAverage

Description

Returns a data set for a moving average on the data set passed in.

Syntax

```
system.stats.getMovingAverage(xData, yData, period)
```

Parameters

Number[] `xData` – An array containing the x-data.

Number[] `yData` – An array containing the y-data.

int `period` – The period to average the data over. Cannot be longer than the length of the dataset.

Returns

double[][] – A 2 dimensional array containing the (x,y) data.

Scope

All

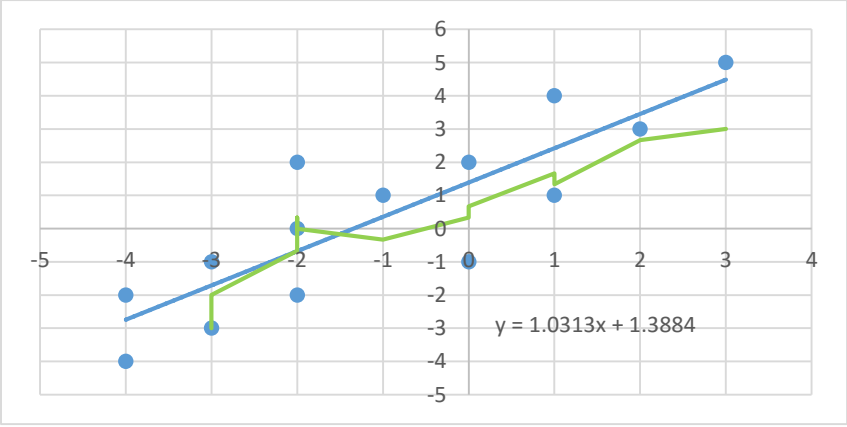
Examples

```
xData = [-4, -4, -3, -3, -2, -2, -2, -1, 0, 0, 1, 1, 2, 3]
yData = [-4, -2, -3, -1, 2, 0, -2, 1, 2, -1, 4, 1, 3, 5]
avgData = system.stats.getMovingAverage(xData, yData, 3)
avgDataString = ''

for data in avgData:
	avgDataString += '(' + str(data[0]) + ',' + str(round(data[1], 2)) + '), '

print avgDataString[:-2]
```

…returns (-3.0,-3.0), (-3.0,-2.0), (-2.0,-0.67), (-2.0,0.33), (-2.0,0.0), (-1.0,-0.33), (0.0,0.33), (0.0,0.67), (1.0,1.67), (1.0,1.33), (2.0,2.67), (3.0,3.0)



Moving average results plotted (in green) over top of regression line (in blue).

system.tag.getAttribute

Description

Gets the value of the specified attribute.

Syntax

```
system.tag.getAttribute(tagpath, attribute)
```

Parameters

String `tagpath` – The path of the tag to get the attribute of.
String `attribute` – The tag attribute to get.

Returns

Object – The value of the specified attribute.

Scope

Client, Designer

Examples

```
print system.tag.getAttribute('analog/lit-100/eu', 'EngHigh')
```

…returns 100.0

system.tag.getChildren

Description

Gets the immediate children under the specified tag path.

Syntax

```
system.tag.getChildren(tagpath)
```

Parameters

String `tagpath` – The path of the location to get the children of. Can be a path to a folder or tag.

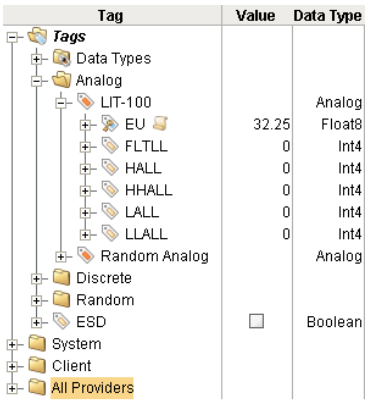
Returns

PyList – A list containing the children.

Scope

Client, Designer

Examples



This snippet returns all items under ‘Tags’

```
print system.tag.getChildren('')
```

…returns ['Discrete', 'Random', 'ESD', 'Analog']

This snippet returns all the tags in the ‘Analog’ folder

```
print system.tag.getChildren('Analog')
```

…returns ['LIT-100', 'Random Analog 1']

This snippet returns all the tags in the LIT-100 UDT

```
print system.tag.getChildren('Analog/LIT-100')
```

…returns ['EU', 'LALL', 'HALL', 'FLTLL', 'LLALL', 'HHALL']

system.template.getFirstValidTemplate

Description

Returns the first template path that matches a template in the current project.

Syntax

```
getFirstValidTemplate(templatePath1, templatePath2, …)
```

Parameters

String `templatePath1` – A path that might be a valid template.

String `templatePath2` – Another path that might be a valid template.

…

String `templatePathN` – Another path that might be a valid template.

Returns

String – The first template path that is an actual template, or null if no template path matched.

Scope

Client, Designer

Examples

This code snippet looks for the first match of three possible templates.

```
getFirstValidTemplate("Tanks/Heated", "Tanks/Cooling", "Tanks/Default")
```

…returns "Tanks/Cooling" because "Tanks/Heated" doesn't exist.

system.template.isValidTemplatePath

Description

Determines if the provided path matches a template in the current project.

Syntax

```
isValidTemplatePath(templatePath)
```

Parameters

String `templatePath` – A path that might be a valid template.

Returns

boolean – True if the template path matches a template in the current project.

Scope

Client, Designer

Examples

This code snippet determines if “Tanks/Heated” is a template.

```
isValidTemplate(“Tanks/Heated”)
```

…returns True because “Tanks/Heated” does exist.

system.window.getWindowInstance

Description

Gets the specified window instance if it is open, and if it was opened with system.window.openWindowInstance().

Syntax

```
system.window.getWindowInstance(window, instanceId)
```

Parameters

String `window` – The name of the window.

String `instanceId` – The instance id of the specific window.

Returns

FPMIWindow – A reference to the specified window.

Scope

Client, Designer

Examples

This snippet **c**hecks if a window opened with system.window.openWIndowInstance() is open.

```
window = system.window.getWindowInstance('Motor', 'P-138')
if window != None:
        print 'Motor P-138 is open'
```

…returns ‘Motor P-138 is open'

system.window.openWindowInstance

Description

Opens a specific instance of a window. If the window with the specified instanceId is already open, it will gain focus. If it is not open, it will be opened and the instanceId will be assigned to an internal property. If parameters are passed, they will be set on the window.

Syntax

```
system.window.openWindowInstance(window, instanceId)
```

Parameters

String `window` – The name of the window.

String `instanceId` – The instance id of the window.

Returns

FPMIWindow – A reference to the opened window.

Scope

Client, Designer

```
system.window.openWindowInstance(window, instanceId, params)
```

Parameters

String `window` – The name of the window.

String `instanceId` – The instance id of the window.

PyDictionary `params` – A dictionary of parameters to pass into the window. The keys in the dictionary must match dynamic property names on the target window's root container. The values for each key will be used to set those properties.

Returns

FPMIWindow– A reference to the opened window.

Scope

Client, Designer

Examples

This code snippet opens an instance of the 'Motor' window.

```
system.window.openWindowInstance('Motor', 'P-138')
```

This code snippet opens an instance of the 'Motor' window and sets a custom property called 'tagPath'.

```
system.window.openWindowInstance('Motor', 'P-138', {'tagPath':'Motor/P-138'})
```

system.window.closeWindowInstance

Description

Closes the specified window instance if it is open, and if it was opened with system.window.openWindowInstance().

Syntax

```
system.window.closeWindowInstance(window, instanceId)
```

Parameters

String `window` – The name of the window.

String `instanceId` – The instance id of the specific window.

Returns

nothing

Scope

Client, Designer

Examples

This code snippet closes an instance of the 'Motor' window.

```
system.window.closeWindowInstance('Motor', 'P-138')
```

Expression Functions

The following expression functions are available with the module.

getFirstValidTemplate

Description

Returns the first template path that matches a template in the current project.

Syntax

```
getFirstValidTemplate(templatePath1, templatePath2, …)
```

Parameters

String `templatePath1` – A path that might be a valid template.

String `templatePath2` – Another path that might be a valid template.

…

String `templatePathN` – Another path that might be a valid template.

Returns

String – The first template path that is an actual template, or null if no template path matched.

Scope

Client, Designer

Examples

This code snippet looks for the first match of three possible templates.

```
getFirstValidTemplate("Tanks/Heated", "Tanks/Cooling", "Tanks/Default")
```

…returns "Tanks/Cooling" because "Tanks/Heated" doesn't exist.

getModuleState

Description

Determines the state of the identified module.

Syntax

```
getModuleState(moduleId)
```

Parameters

String `moduleId` – The identifying string of the module.

Returns

Integer – 1 if active, 0 if expired, -1 if moduleId not found.

Scope

Client, Designer

Examples

This code snippet displays the state of the Vision (previously Factory PMI) module.

```
getModuleState("fpmi")
```

…returns 1 because the module is found and the trial is not expired.