PI Data to Matlab

Structure Documentation

|  |  |
| --- | --- |
| Marissa Engle  OSIsoft, LLC |  |

# Program Structure Overview

The structure of this program centers on Main Control. Main Control connects access to both Matlab and AF based on manipulations of the Main Form. Main Control handles initial setup of the Main Form by accessing the AF database for Elements and PI Points.

AF and Matlab Access contain methods to reach their designated domains. AF Access retrieves data and then passes it on to Matlab Access. AF Access will find elements, search, and get data. Matlab Access makes sure variable names aren’t overwritten and are valid Matlab variable names as well as packages the data to pass to Matlab.

Log System represents a list of data transactions from AF to Matlab that the user has made. It allows for users to save their work and then reimport it in based on a formatted text file.

# Main Control

The Main Control maintains the connection between Matlab and AF. It also handles saving and importing of log files.

## Constructor (Main Control)

Opens the connection to the Matlab through MatlabAccess.MatlabStartup().

## Methods

### setMainForm

Parameters: **MainForm form** – Main view Form

Assigns the Main Form to Main Control on its construction.

### checkMatlab

Parameters: **bool showDialog**

Maintains an open Matlab connection. If Matlab is closed unexpectedly or on purpose, the user can reopen Matlab. Matlab can be reopened with logged inputs, empty, or left closed, but a log of data transactions will still be maintained. ShowDialog controls the dialog so that it is only shown once for each time that Matlab goes down.

### getAFDatabase/setAFDatabase

Gets/Sets the AFDatabase from the MainControl to keep the Main Form View and Main Control in sync.

### getAttributes

Parameters: string path – Element Path

Accesses the AFDatabase for Attributes using the Element Path.

### getPIPoints

Parameters: **PISystem sys**, **PIServer serv**

Accesses the PIServer to get the PIPoints that are available.

### getPIData

Parameters: **string point** – name of the point

**String server** – The name of the PIServer

**String MatlabName** – The name of the variable used in Matlab

**String start** – the string representing start time (Relative or Absolute)

**String end** - the string representing end time (Relative or Absolute)

**bool edit** – whether the log input is edited or not.

Checks the new addition against Matlab Variable Names. Access the AF Database for the data from the Pi point.

### getAFData

Parameters: **string server-database** – name as a string ‘server’-‘database’

**String attName** – name of the attribute

**String MatlabName** – name of the attribute in Matlab

**String searchPath** – Element Path

**String start** – absolute or relative start time

**String end** – absolute or relative end time

**Bool edit** – whether the log input is edited or not.

Gets the AF data for a non PI Point attribute. Checks that the Matlab Name is not used already in Matlab. Uses AF Access to reach the data.

### importLog

Parameters: **string path** – path to the imported file

Imports a file previously saved. It contains a structured log input file so that all values can be easily imported into Matlab.

### ImportToAF

Parameters: **string path** – Element Path

**String workspaceVariableName** – the name of the variable in the Matlab workspace

**String AFName** – the name of the attribute in AF being written to.

Allows writing single data points to an AF attribute.

### saveCurrentLog

Parameter: **string filePath** – where the log file is saved.

Saves a formatted text file that can be imported for quick loading of data into Matlab.

### currentDataPrefChanges

Parameter: **bool Timestamp** – the timestamps are included in the export to Matlab

**Int datapref** – integer referring how the data is reported.

**String format** – a formatted string representing the users choices.

Sets the data preferences that are used for the data acquisition from AF Database. The data preference give the choice of raw data (0), sampling data (negative), and profiled sampling data (positive). The magnitude of the integer is a guideline for how many data points should be returned.

# AF Access

## Grouped methods for accessing the AF database data and allowing for PI Time as well as dates to be used.

### checkAFTimeRange

Parameters: **string start** – string representing the start time

**String end** – string representing the end time

Validates the start and end times, by creating an AFTimeRange and returning it.

### isAbsoluteTimeString

Parameters: **string start** – string of start time

**String end**  - string of end time

**LogInput loginput** – Log input object containing information on the data transaction

Sets the log input variable absolute time to have an array containing the start and end time.

### setDataPrefs

Parameters: **bool Timestamp** – the timestamps are included in the export to Matlab

**Int datapref** – integer referring how the data is reported.

**String format** – a formatted string representing the users choices.

Sets the data preferences that are used for the data acquisition from AF Database that are held in AF Access. The data preference give the choice of raw data (0), sampling data (negative), and profiled sampling data (positive). The magnitude of the integer is a guideline for how many data points should be returned.

### getAFData

Parameters: **string server-database** – name as a string ‘server’-‘database’

**String attName** – name of the attribute

**String MatlabName** – name of the attribute in Matlab

**String searchPath** – Element Path

**String start** – absolute or relative start time

**String end** – absolute or relative end time

**Bool addToListView** – whether the log input is edited or not.

Gets the AF data for a non PI Point attribute.

### getEventFrameData

Parameters: **string server\_database** – name as a string ‘server’ – ‘database’

**String AttributeName** – name of the attribute

**String MatlabName** – name of the attribute in Matlab

**AFEventFrame frame** – the AF Eventframe object

**Bool addToListView** – whether the log input is edited or not.

Gets the data from an EventFrame.

### getPIData

Parameters: **string server** – name as a string PIServer ‘server’

**String MatlabName** – name of the attribute in Matlab

**String point** –name of the PI Point

**String start** – absolute/relative start time

**String end** – absolute/relative end time

**Bool addToListView** – whether the log input is edited or not.

Gets the data from a PI Point.

### getData

Parameters: **string server-database** – name as a string ‘server’-‘database’

**String AttributeName** – name of the attribute

**String MatlabName** – name of the attribute in Matlab

**String start** – absolute or relative start time

**String end** – absolute or relative end time

**Object AFObject** – Object containing an AFObject (EventFrame, Element, or PI Point)

**Bool addToListView** – whether the log input is edited or not.

Generic Method for getting the data from the AF Database.

### AFValuesToArray

Parameters: **array** – array of values from ConvertAFValues.GetValuesArray()

Packages the data from AF into an array for a single delivery into Matlab.

### writeToAF

Parameters: **AFElement element** – AF element

**AFAttribute attribute** – AF attribute

**Double value** – value being imported into AF

Imports a new value for the passed-in element and attribute and checks in the new change into the database.

# Matlab Access

## Grouped methods for accessing the Matlab program and data and allowing for the modification and addition of Matlab variables into the Matlab Workspace. It also protected imported or existing variable names.

## MatlabStartup

Initiates the Matlab connection.

1. Tries to connect to an open Matlab (this has to be enabled as an Automation Server with enableservice(‘AutomationServer’,true))
2. Otherwise opens a new Matlab window already enabled.
3. If Matlab does not exist or cannot be opened you can run the program without Matlab.

## isMatlabOpen

Checks that Matlab has not been closed when the user tries to add data.

## isVariableInWorkspace

Checks if Matlab workspace has the variable name already.

## modifyMatlabName

It changes the user input into a valid Matlab variable name by removing punctuation and spaces. It also adds an integer to the name that it iterates through until the Matlab name is unique.

## sendDataToMatlab

Sends double values to Matlab.

## sendDataToMatlab

Sends string values to Matlab.

## GetWorkspaceData

Gets a workspace variable value for import into the AF Database.

## removeMatlabVariable

Removes the variable from the Matlab workspace. Only occurs when the log input variable name is changed. This also removes the timespace data as well.

## plotMatlabVariables

Example of executing Matlab Calls from C#. It can plot the variable against time in Matlab.

# Log System

Records the data transactions that a user has conducted while using the utility. When imports from files are done, the imported data is also recorded. This allows for the modification and saving of a text file that can be imported later.

## Clear

Clears the Data Transaction History.

## getLogInput

Parameter: **string keyVariableName** – uses unique variable name to get Log Input from map

Gets the Log Input Object using the variable name.

## addLogInput

Parameter: **string keySrv\_Db** – the key representing the server and db.

**LogInput logInput** – the object of the data transaction

**Bool addToListView** – generally true, it is false when editing.

Adds a Log Input, or object representing a single transaction to the Log. First checks that the server and database key exists, adds the variable to the server list, and then adds it to the ListView.

## removeLogInput

Parameter: **string keyName** – name of the Log Input variable

**string srv\_db** – the key representing the server and db.

Removes a log input from the logged transaction list, but does not change the Matlab Workspace.

## parseServerDatabaseKey

Parameter: **string srv\_db** – the key representing the server and db.

Parses the srv\_db string into the server name and the database name.

## SaveLog

Parameter: **string Filename** – path to where file will be saved

Saves the transaction log in a file \*.txt.

## ImportLog

Parameter: **string Filename** – path to the text file that is imported.

Imports in a text file that is in the correct format.

## isLogEmpty()

Checks to see if no data transactions have occurred.

## isVariableInLogSystem

Parameter: **string varname** – Variable name being checked

Check if the variable name has already been used and is thus not unique.

## isServer\_DatabasesInLogSystem

Parameter: **string srv\_db\_name** – Server Database key checked for.

Checks to see if a server database key exists.

## addView

Parameter: **ListView lv** – Listview that shows data transactions.