Simio
Forward Thinking

Simio API Note: MATLAB Custom Steps

Creation: July 2020 (Dan Houck)

Special thanks to Dr. Mohammed Dehghani of NorthEastern U. for the original source code.

Contents

Simio
Forward Thinking

Overview

This API Note describes how a Simio User-Defined Step can be created to communicate with MATLAB. The original example was derived with permission from the source of Dr. Mohammed Dehghani.

This Note describes some complex programming topics. It assumes that the reader is familiar with C#, MATLAB, and .NET technologies, and that the user has access to a valid MATLAB license.

Before running these examples, the MLApp COM Process from MATLAB must be installed. Please refer to the appendix for instructions on how to do this correctly.

Some Background Information on MATLAB

What is MATLAB?

From Wikipedia: “MATLAB (matrix laboratory) is a multi-paradigm numerical computing environment and proprietary programming language developed by MathWorks.”

Although originally designed for mathematical operations, it has several add-ons that provide capabilities for machine-learning, robotics, control systems, and other functions.

The company website is here: https://www.MathWorks.com

Simio
Forward Thinking

MATLAB User Step Code

This API Note provides multiple MATLAB steps, which are expected to grow over time. At the core of each is a basic template which involves calling into the MATLAB application and using the MATLAB language. At the time of this writing there are:

1. Play a sound wave file.
2. To be Determined

MATLAB Step Template

All the MATLAB steps also uses a singleton structure to gain access to the MATLAB application (MLApp). Invoking the MLApp is time consuming, as it is a Windows COM Server, so the singleton pattern allows this to be done only once.

For this to work, the MLApp COM Server must be correctly installed. See the Appendix “Installing the COM Interface to MATLAB” for instructions on how to do this.

The overall organization for a Simio User-Define MATLAB steps is this:

Define the Simio Step definitions so that from within Simio design you can set parameters. For example, the folder and filename properties for a sound file.

Define what happens when the Step is initialized. For example, retrieving the properties that are used when the step is executed, and calling the Matlab context once so that the singleton can store the MATLAB MLApp.

Define what happens each time the Step executes. For example, check the value of a Simio State variable to determine which sound file is to be executed.

By convention, we are going to call the step Matlab<step-function>Step, so our first example step that plays sound files is MatlabPlaySoundStep.

There is a shared MATLAB library called MatlabHelpers that contains code common to all steps, such as logging (tracing).

Simio
Forward Thinking

MATLAB Step: MatlabPlaySoundStep

The MatlabPlaySoundStep is the “Hello World” step for the MATLAB examples. It is a variation of the example provided by Dr. Mohammed Dehghani.

The Step requires a path to a folder where the MATLAB function files reside, as well as a path to sound files.

When the step is executed, the designated file(s) are played.

This uses the Simio project file ModelTestMatlab1.spfx which is set up to use the following folders:

C:\Test\MatlabFiles – Holds the MATLAB files (e.g. PlaySoundFile.m)

C:\Test\SoundFiles – Holds the sound files (*.wav and *.mp3)

Simio C# Step Code

All the C# code is contained in the MatlabPlaySoundStep.cs file.

The DefineSchema function creates string properties for:

SoundFilePath

MatlabFolderPath

During execution, the CallMatlabPlaySoundFile method is called.

That routine is very simple. After checking for valid arguments, it sets the directory to the MATLAB folder, and then constructs a MATLAB command that is a call to the PlaySoundFile function with the argument being the path to the sound file.

Finally, it looks at the result, which MATLAB formats with some formatting characters (such as CR, AKA “carriage-returns”) and “ans =” and parses out the result.

Simio
Forward Thinking

MatlabSteps* MatlabHelpers.cs MatlabPlaySoundStep.cs

MatlabSteps MatlabSteps.MatlabPlaySoundStep

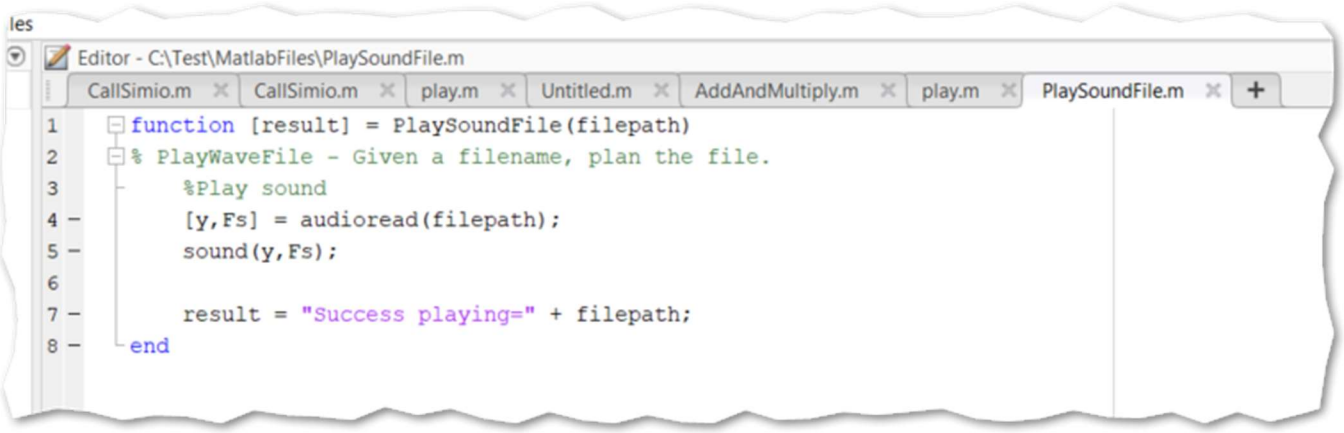
```
176             // Check for folder and file
177             if (!File.Exists(soundFilePath))
178             {
179                 explanation = $"Cannot find File={soundFilePath}";
180                 return false;
181             }
182
183             marker = "Setting directory location for MATLAB files";
184             // Call a MATLAB command to change the file location to where the file that holds the function is located.
185             string cmd = $@"cd {matlabFolder}";
186             matlab.Execute(cmd);
187
188             marker = "Calling the MATLAB function";
189             // Build the MATLAB command
190             string matlabCommand = $@"PlaySoundFile(""{soundFilePath}"")";
191
192             marker = "Getting the result";
193             // A successful answer (when trimmed) starts with "ans = " followed by whatever the function return.
194             string result = matlab.Execute(matlabCommand);
195
196             if ( !result.Trim().StartsWith("ans ="))
197             {
198                 explanation = $"MATLAB Failure. SoundFile={soundFilePath} Result={result.Trim()}";
199                 return false;
200             }
201             else
202                 return true;
203
204         }
```

Simio
Forward Thinking

MATLAB Code

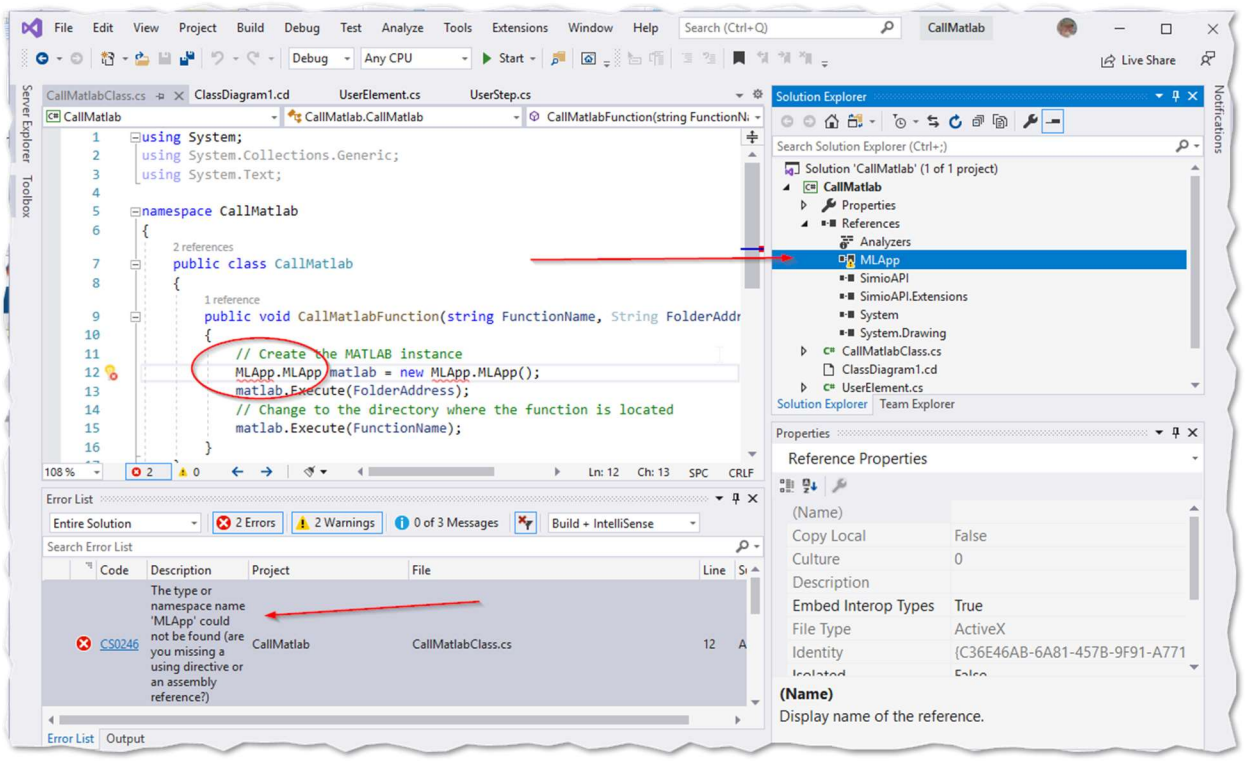
The MATLAB function is in a file by the same name within the folder that the C# code pointed to.

The code is exceptionally simple:



Simio
Forward Thinking

Appendix – Installing the COM Interface to MATLAB

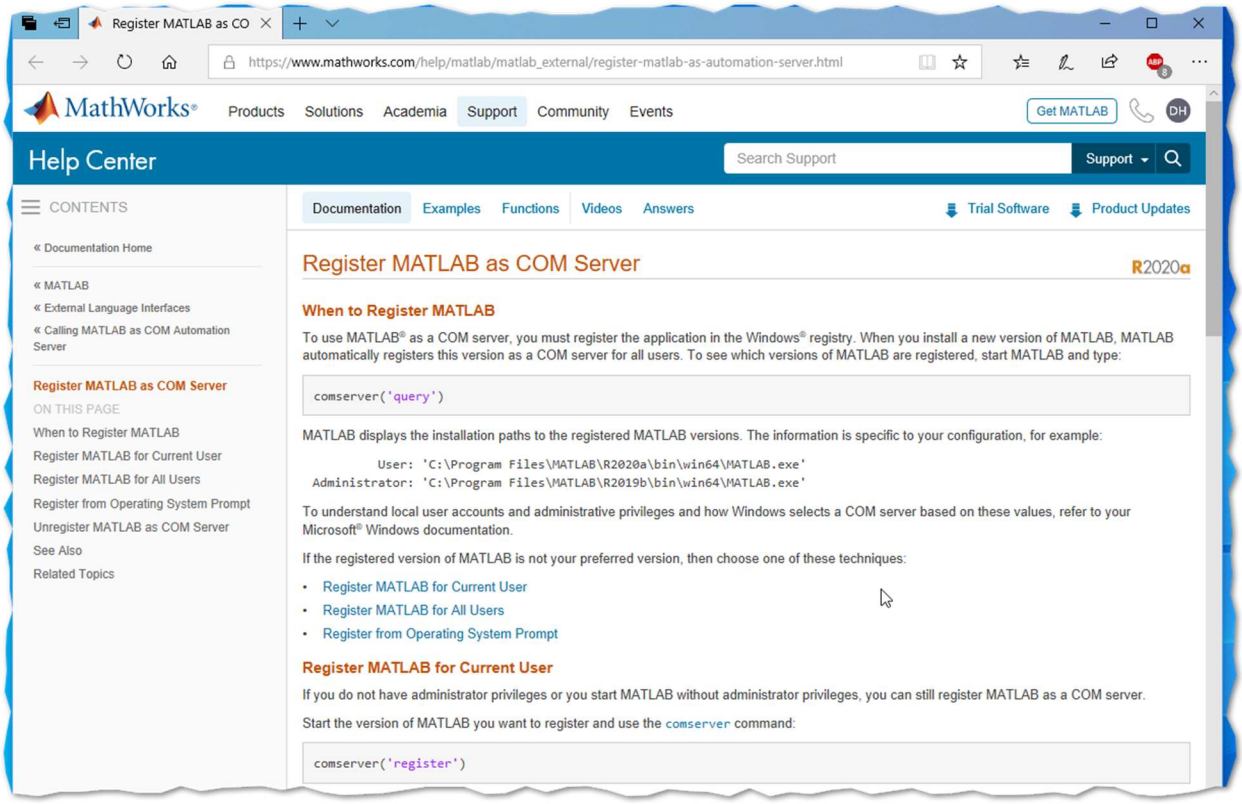


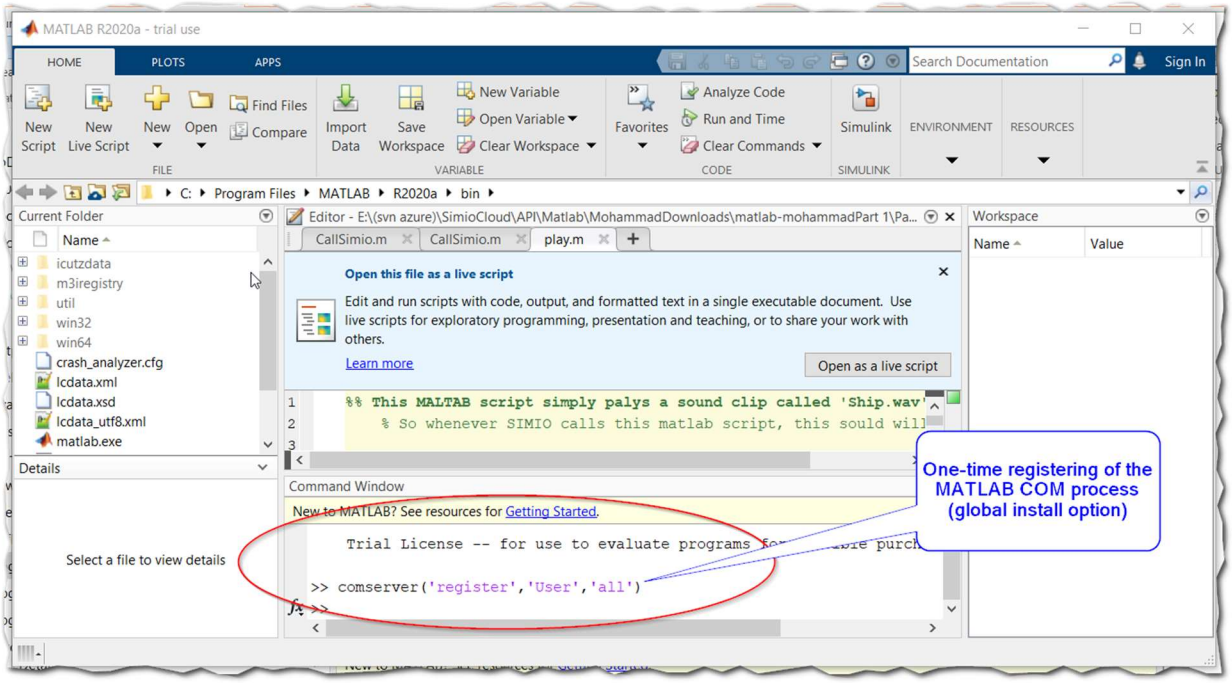
Here is the MATLAB link:

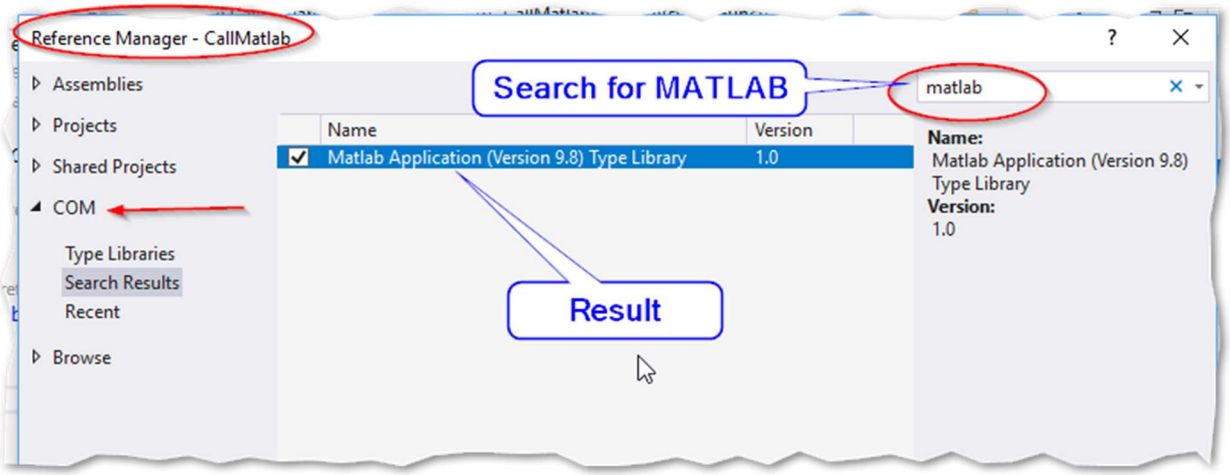
https://www.mathworks.com/help/matlab/matlab_external/register-matlab-as-automation-server.html

Which should bring you to something that looks like this:

Simio
Forward Thinking

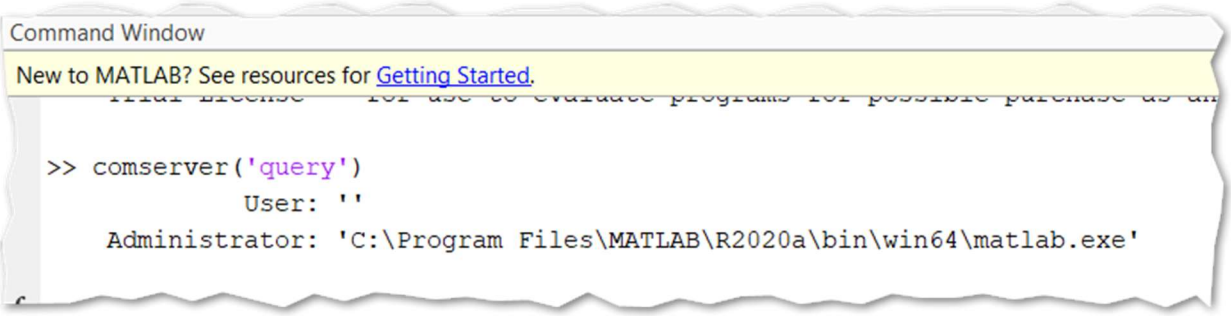






To find which version of MATLAB you are communicating with, type:

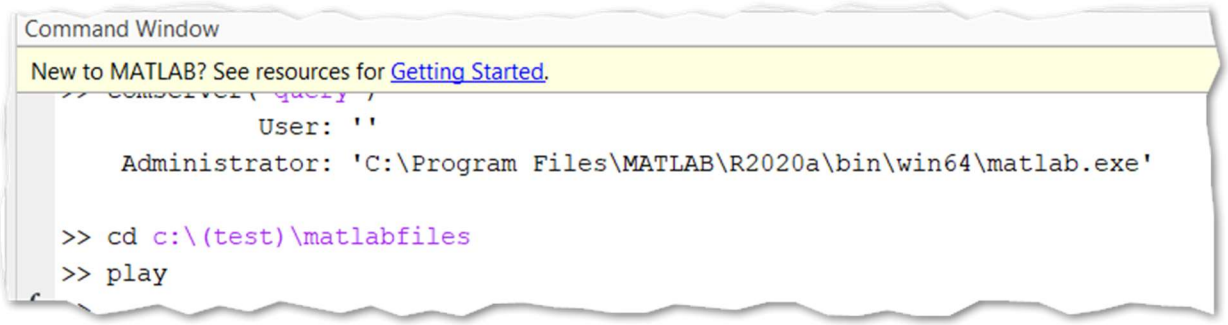
comserver('query'), and the reply should look something like:



Note: the comserver is available for MATLAB R2020a and later. For previous versions, consult your documentation.

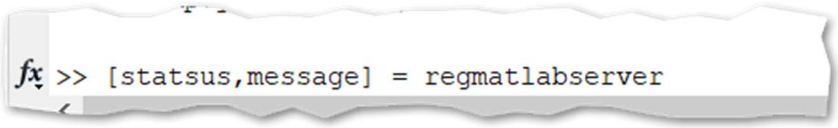
From this command line, you can also test the methods you wish to execute in the Simio Step. For example:

Simio
Forward Thinking



You must also register MATLAB as an Automation Server.

This is done by issuing the regmatlabserver command:



Tips:

MATLAB commands are case-sensitive!