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

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

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).

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:

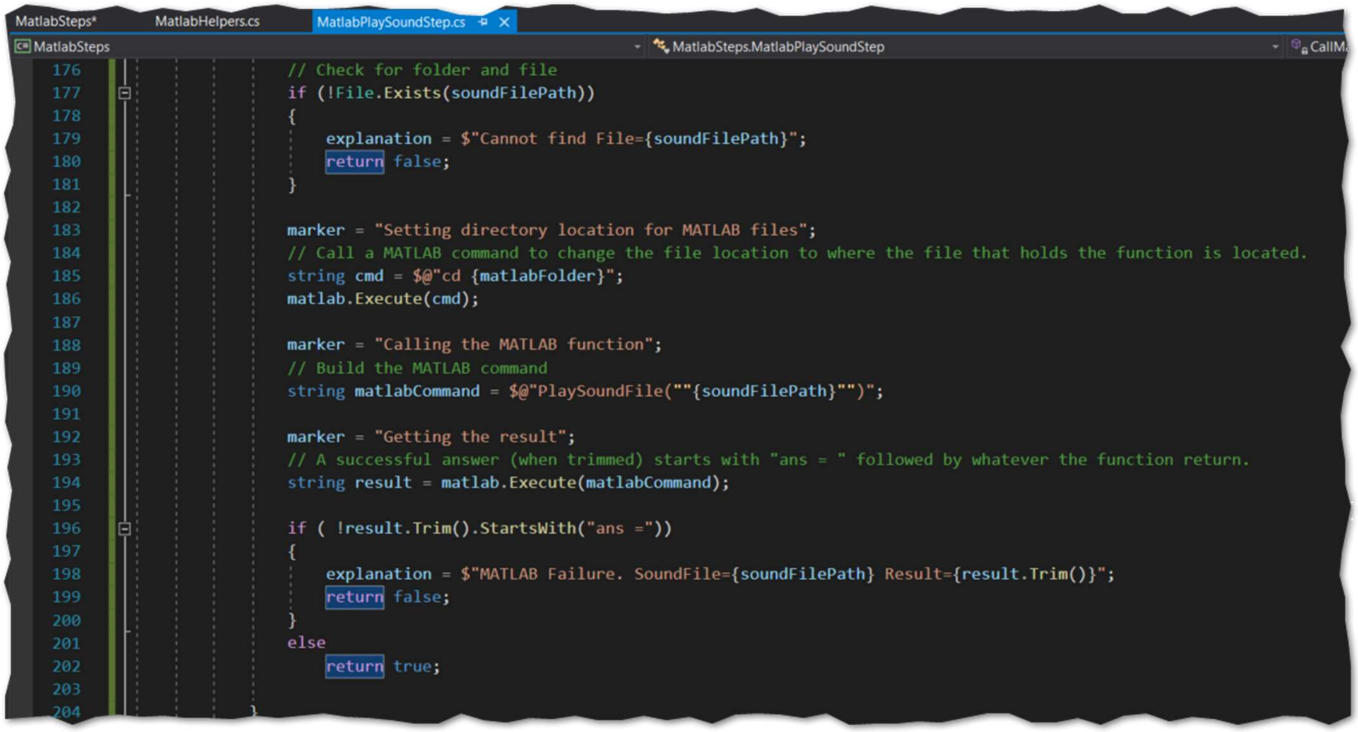
SoundFilePath

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 returns and “ans =” and parses out the result.

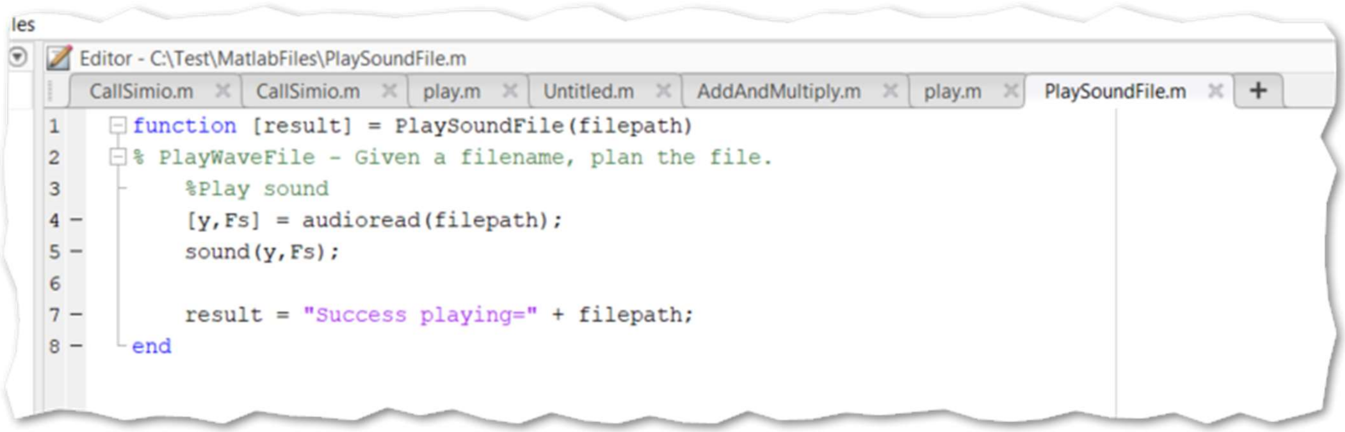


```
                // Check for folder and file
                if (!File.Exists(soundFilePath))
                {
                    explanation = $"Cannot find File={soundFilePath}";
                    return false;
                }

                marker = "Setting directory location for MATLAB files";
                // Call a MATLAB command to change the file location to where the file that holds the function is located.
                string cmd = $@"cd {matlabFolder}";
                matlab.Execute(cmd);

                marker = "Calling the MATLAB function";
                // Build the MATLAB command
                string matlabCommand = $@"PlaySoundFile(""{soundFilePath}"")";

                marker = "Getting the result";
                // A successful answer (when trimmed) starts with "ans = " followed by whatever the function return.
                string result = matlab.Execute(matlabCommand);

                if ( !result.Trim().StartsWith("ans ="))
                {
                    explanation = $"MATLAB Failure. SoundFile={soundFilePath} Result={result.Trim()}";
                    return false;
                }
                else
                    return true;

            }
```

MATLAB Code

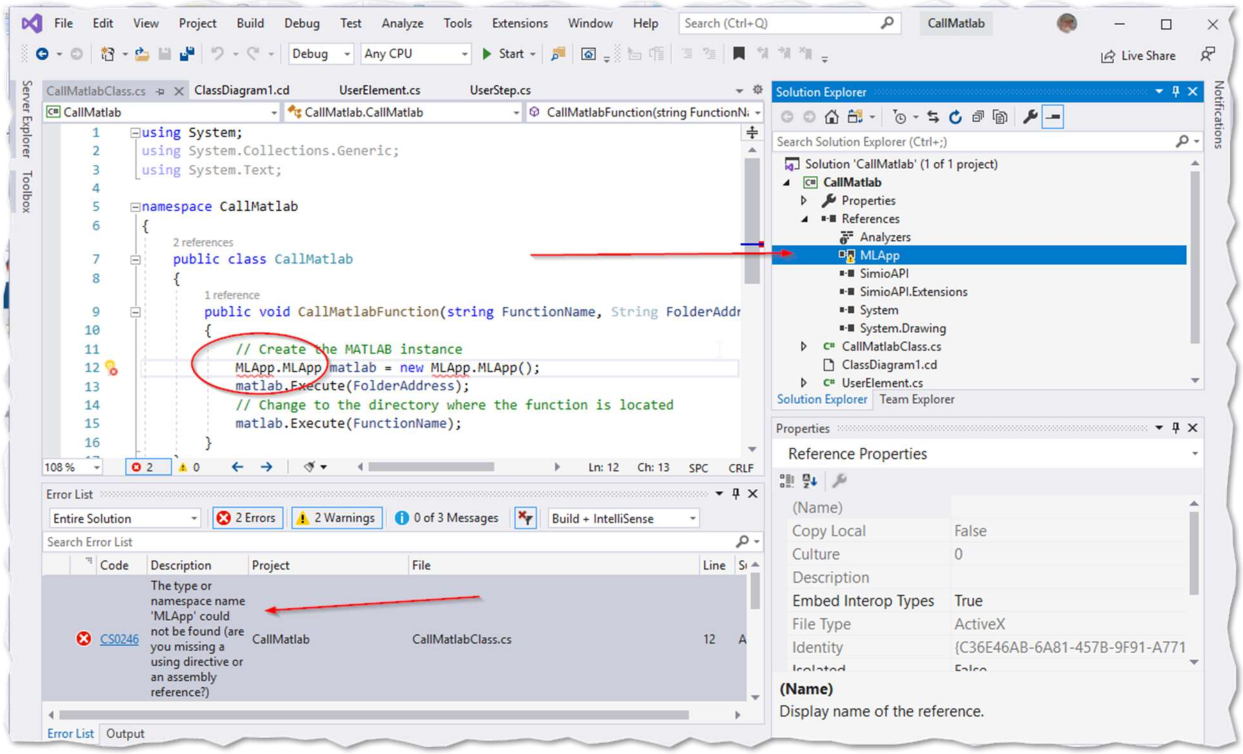
The MATLAB function is in a file by the same name within the folder that the C# code pointed to.

The code is exceptionally simple:



```
function [result] = PlaySoundFile(filepath)
% PlayWaveFile - Given a filename, plan the file.
    %Play sound
    [y,Fs] = audioread(filepath);
    sound(y,Fs);

    result = "Success playing=" + filepath;
end
```

Appendix – Installing the COM Interface to MATLAB



```
using System;
using System.Collections.Generic;
using System.Text;

namespace CallMatlab
{
    2 references
    public class CallMatlab
    {
        1 reference
        public void CallMatlabFunction(string FunctionName, String FolderAddr
        {
            // Create the MATLAB instance
            MLApp.MLApp matlab = new MLApp.MLApp();
            matlab.Execute(FolderAddress);
            // Change to the directory where the function is located
            matlab.Execute(FunctionName);
        }
```

Error List: Entire Solution | 2 Errors | 2 Warnings | 0 of 3 Messages | Build + IntelliSense

Code	Description	Project	File	Line
CS0246	The type or namespace name 'MLApp' could not be found (are you missing a using directive or an assembly reference?)	CallMatlab	CallMatlabClass.cs	12

Solution 'CallMatlab' (1 of 1 project)
- CallMatlab
  - Properties
  - References
    - Analyzers
    - MLApp
    - SimioAPI
    - SimioAPI.Extensions
    - System
    - System.Drawing
  - CallMatlabClass.cs
  - ClassDiagram1.cd
  - UserElement.cs

Properties: Reference Properties

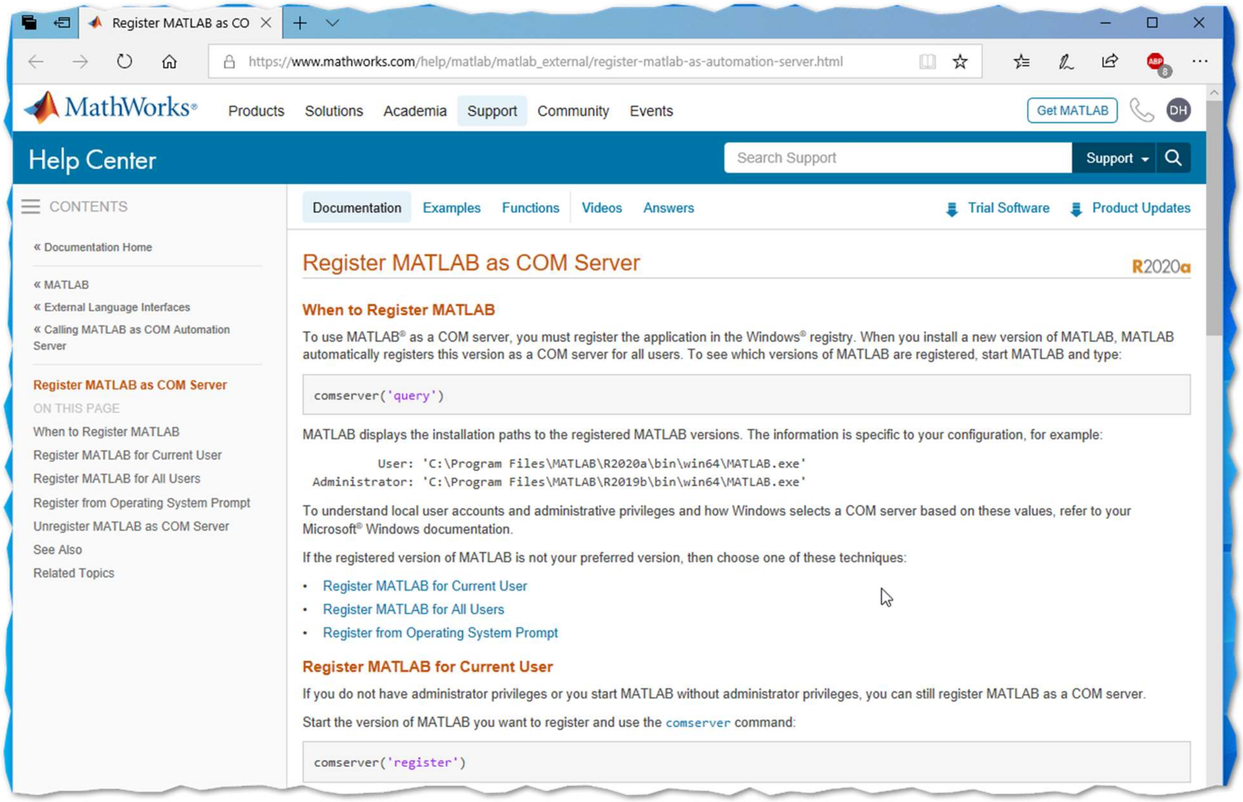
(Name)	
Copy Local	False
Culture	0
Description	
Embed Interop Types	True
File Type	ActiveX
Identity	{C36E46AB-6A81-457B-9F91-A771
Isolated	False

(Name)
Display name of the reference.

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:



Register MATLAB as CO
https://www.mathworks.com/help/matlab/matlab_external/register-matlab-as-automation-server.html
MathWorks
Products
Solutions
Academia
Support
Community
Events
Get MATLAB
Help Center
Search Support
Support
CONTENTS
Documentation
Examples
Functions
Videos
Answers
Trial Software
Product Updates
« Documentation Home
« MATLAB
« External Language Interfaces
« Calling MATLAB as COM Automation Server
Register MATLAB as COM Server
ON THIS PAGE
When to Register MATLAB
Register MATLAB for Current User
Register MATLAB for All Users
Register from Operating System Prompt
Unregister MATLAB as COM Server
See Also
Related Topics

Register MATLAB as COM Server

R2020a

When to Register MATLAB

To use MATLAB® as a COM server, you must register the application in the Windows® registry. When you install a new version of MATLAB, MATLAB automatically registers this version as a COM server for all users. To see which versions of MATLAB are registered, start MATLAB and type:

```
comserver('query')
```

MATLAB displays the installation paths to the registered MATLAB versions. The information is specific to your configuration, for example:

```
         User: 'C:\Program Files\MATLAB\R2020a\bin\win64\MATLAB.exe'
Administrator: 'C:\Program Files\MATLAB\R2019b\bin\win64\MATLAB.exe'
```

To understand local user accounts and administrative privileges and how Windows selects a COM server based on these values, refer to your Microsoft® Windows documentation.

If the registered version of MATLAB is not your preferred version, then choose one of these techniques:

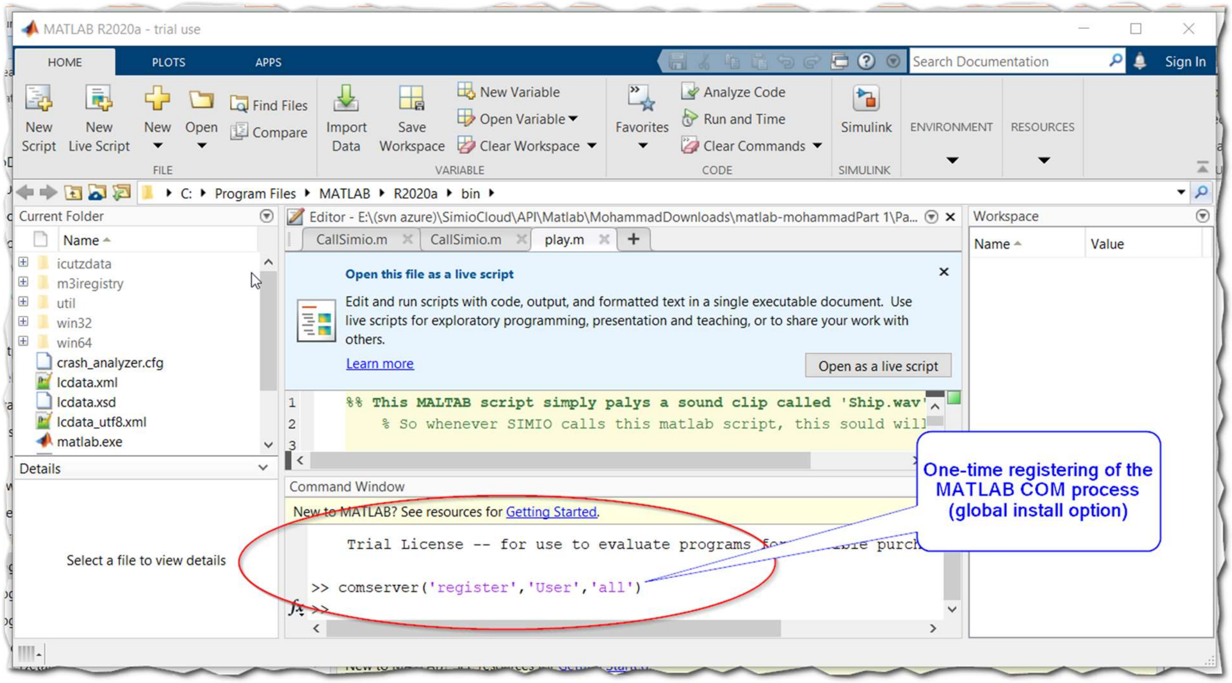
- Register MATLAB for Current User
- Register MATLAB for All Users
- Register from Operating System Prompt

Register MATLAB for Current User

If you do not have administrator privileges or you start MATLAB without administrator privileges, you can still register MATLAB as a COM server.

Start the version of MATLAB you want to register and use the `comserver` command:

```
comserver('register')
```

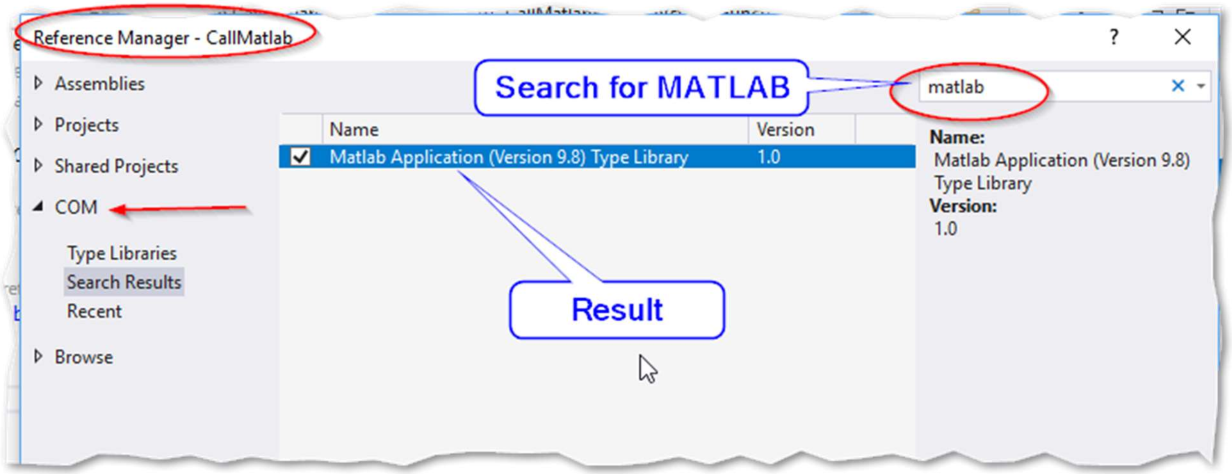


```
%% This MALTAB script simply palys a sound clip called 'Ship.wav'
   % So whenever SIMIO calls this matlab script, this sould will
```

New to MATLAB? See resources for Getting Started.

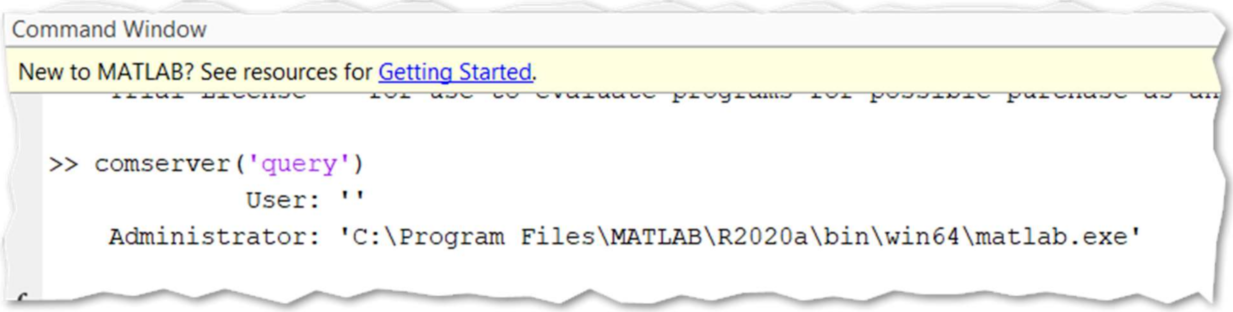
```
  Trial License -- for use to evaluate programs for [illegible] purch

>> comserver('register','User','all')
>>
```

One-time registering of the MATLAB COM process (global install option)



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

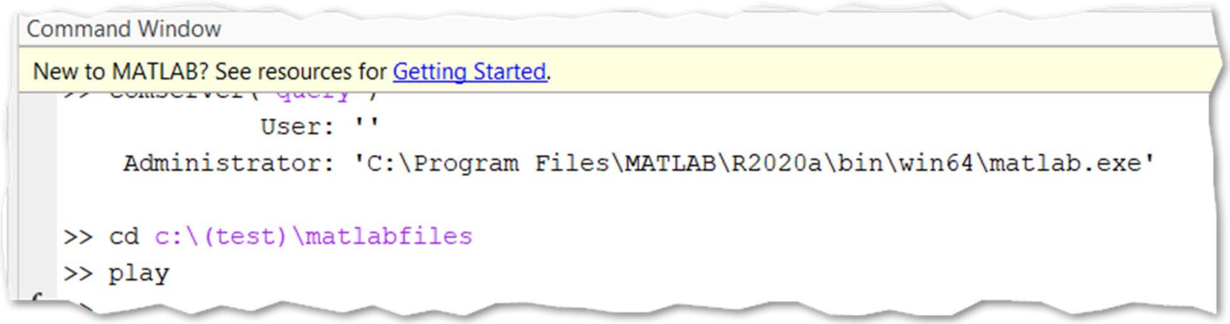
comserver('query'), and the reply should look something like:



```
>> comserver('query')
            User: ''
   Administrator: 'C:\Program Files\MATLAB\R2020a\bin\win64\matlab.exe'
```

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:



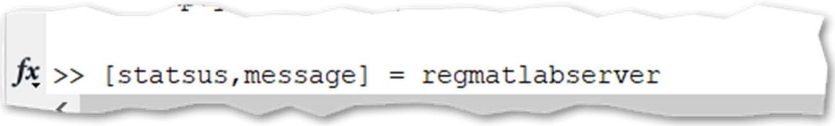
Command Window

New to MATLAB? See resources for Getting Started.

```
>> comserver('query')
              User: ''
     Administrator: 'C:\Program Files\MATLAB\R2020a\bin\win64\matlab.exe'

>> cd c:\(test)\matlabfiles
>> play
```

You must also register MATLAB as an Automation Server.

This is done by issuing the regmatlabserver command:



```
>> [statsus,message] = regmatlabserver
```

Tips:

MATLAB commands are case-sensitive!