

Simio Solutions Series



Using the Python Scripted Step

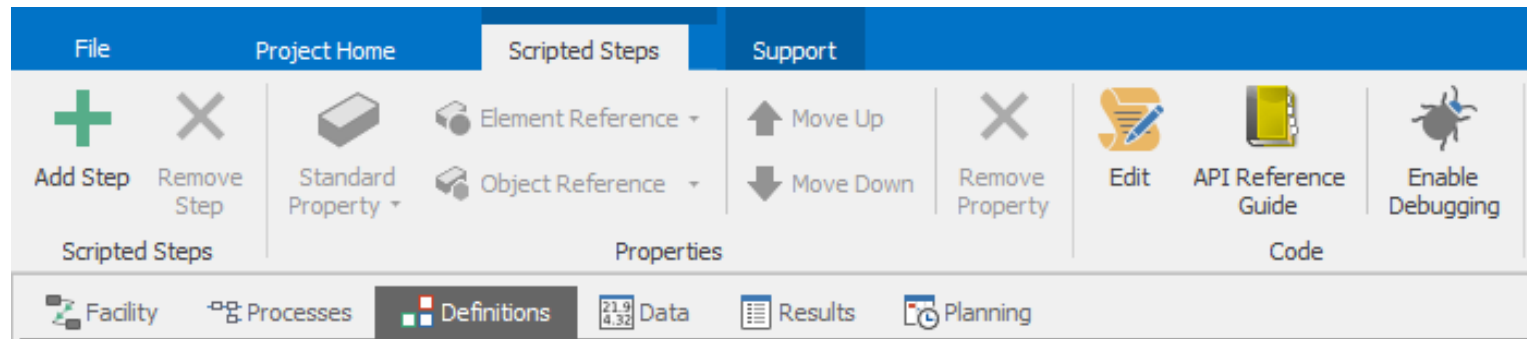
October 22, 2025

Jeff Smith

jsmith@simio.com

Presentation Objectives

- Understand Simio's new Python Scripted Step – how it works and some of the basic features.
- See the Python Scripted Step in action through several examples



- **Note** – this is an advanced feature – We assume that you already know how to use Simio *Processes* and that you know Python.

Python Scripted Step Template

```
from SimioAPI import *
from SimioAPI.Extensions import *

# called when a process token executes the step.
def Execute(propertyReaders: IPropertyReaders, context: IStepExecutionContext)
    -> ExitType:
    context.ExecutionInformation.TraceInformation("Step: MyStep1 executed")
    # ...
    return ExitType.FirstExit
```

Simio Help

Simio Reference Guide


Scripted Steps - Discussion and Examples


Some of your Simio model is and/or can be exposed to your Python step. This is done in two ways and follows the same convention as the Simio API reference guide:

1. A "context" is passed every time is shorthand for the interface name, "IStepExecutionContext". Note, this is searchable in the Simio Reference API Guide found in the installation directory of Simio. In short, it exposes a small set of model level and step level functions. From this you can do things like log out to Trace window and store an object of data for use later or retrieve a previously stored object of data by a user friendly name of your choosing.
2. A "property reader". Your scripted step only has data from the model you explicitly pass in. You are only able to update a property or state from your Python step if you pass it in. Property Reader is the bridge to one-to-as-many values you choose to expose to Python scripted step. Because this listing can change over time as you develop your Python step, you first are asked to find the value in that list and cast it to the right type in Python. This assists with auto-complete as well for easier programming. For common casting of properties, please reference the [GitHub repo examples](#).

Example Models – Simio GitHub Site

- <https://github.com/SimioLLC/PythonExamplesInSimio>


 01 Python Integration - Test Initial Python Configuration Model.docx

 02 Python Integration - General States and Properties.docx

 03 Python Integration - Data Tables with State Columns.docx


 04 Python Integration - Write to Output Table.docx


 05 Python Integration - Server Selection Model.docx


 06 Python Integration - Pandas DataFrames From Tables.docx


 07 Python Integration - Using ModelGlobalData.docx


 08 Python Integration - Testing Shared Files.docx


 Callback Test V01.spfx


 Data Tables With State Columns V01.spfx


 General States and Properties V01.spfx


 Pandas DataFrames from Tables V01.spfx

 Server Selection V01.spfx

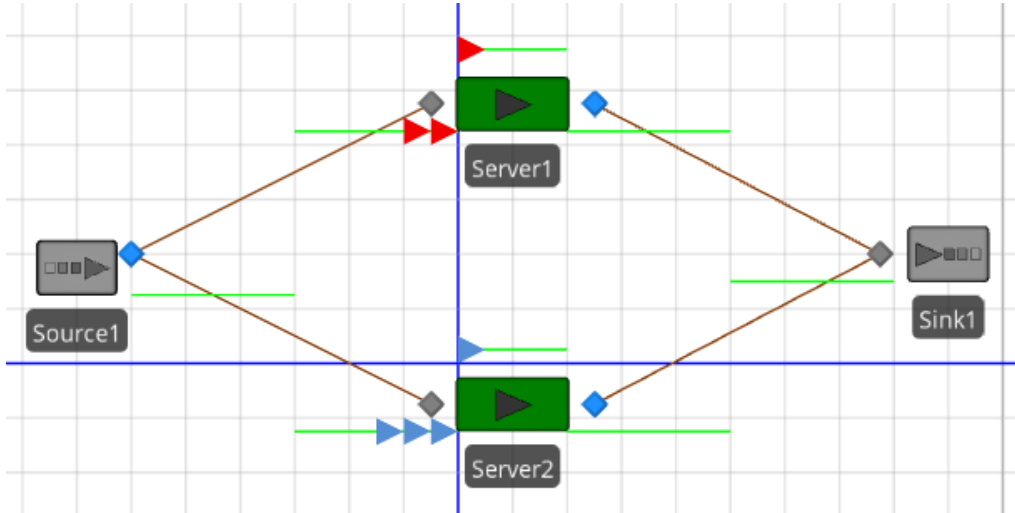
 Test Initial Python Configuration V01.spfx

 Testing Shared Files V01.spfx

 Using ModelGlobalData V01.spfx

 Write to Output Table V01.spfx

Example – Server Selection



Simio Expressions:

```
Server1.InputBuffer.Contents  
Server2.InputBuffer.Contents  
Token.ReturnValue  
ModelEntity.Picture
```

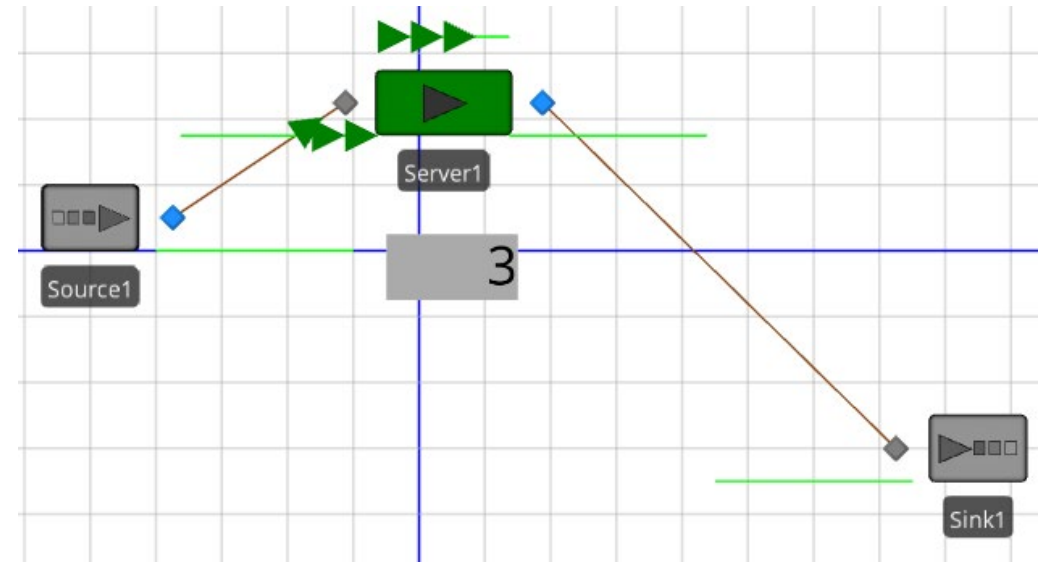
- Goal – When an entity arrives, select the server with the shorter queue, defaulting to a preferred server if the queue lengths are equal.
- Information needed:
 - Server 1 Queue Length
 - Server 2 Queue Length
 - Preferred Server

} *step properties*
- Implement selection
 - Selected server
 - Change entity symbol

} *states*

Example – Interacting with a Server Object

- Multiple Scripted Steps
 - One used in OnRunInitialized
 - Another used in the Process add-on process.
- Passing an object reference property and interacting with the object in the Python code
- Using `print()` to add debugging messages
- Interactive debugging using VS Code



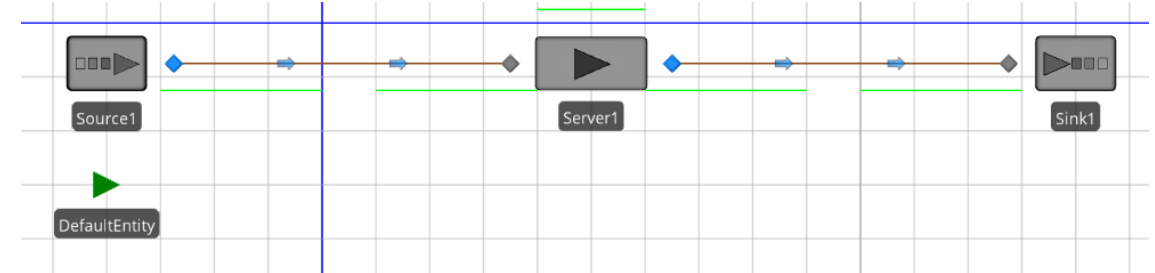
Using the print() function

- The standard Python print() function prints the specified string to SimioActions.log
 - Default path: C:\Users*user_name*\AppData\Local\Simio LLC\Simio\SimioActions.log
- I use a helper function that adds a searchable tag to the print string (lots of other stuff is printed to the log):

```
def lprint(str):  
    print(f"[SCRIPTPRINT] {str}", flush=True)
```


Example – Using ModelGlobalData

- Using an external Python package (Pandas)
- Reading a Data Table in a Python step
- Using ModelGlobalData to store and retrieve data structures between Python scripts and script executions



Facility Processes Definitions Data Results Planning

Views < Items Orders Order Items

Import: [CSV Data Importer 1], Bound to CSV: items.csv, Use headers = True, Separator = ','

Last import was 160 days, 10 hours, and 15 minutes ago


	Item Number	Item Name	Rack	Node
1	RVE-261-NB4E5	Portable Ice Cubes	r_a7c22b	n_a7c22
2	ZQD-734-4V67B	Anti-Gravity Ring	r_a5c37t	n_a5c37
3	WJT-389-Y9R3V	Enchanted Clothes	r_a3c35b	n_a3c35
4	SPU-219-2D10O	Inflatable Bottle	r_a7c34b	n_a7c34
5	EHR-148-4360B	Enchanted Dog Leash	r_a7c9t	n_a7c9
6	FFN-681-RDM41	Self-Untying Teacup	r_a6c47b	n_a6c47
7	STJ-486-QQJ1N	Edible Storybook	r_a1c13t	n_a1c13
8	QMO-930-BLTSK	Anti-Gravity Teacup	r_a7c29b	n_a7c29
9	XRV-549-NA2D3	Disappearing Flowers	r_a1c6t	n_a1c6
10	NZG-21-NNKGB	Self-Correcting Rainbow	r_a2c39b	n_a2c39
11	DPB-540-2U9ZT	Miniature Hammock	r_a5c41b	n_a5c41
12	YSL-357-SWRMR	Automatic Ring	r_a4c31t	n_a4c31
13	TKD-550-D4SLN	Edible Ice Cream Maker	r_a3c45b	n_a3c45
14	AKF-818-FJRQT	Never-Melting Pillow	r_a4c38t	n_a4c38
15	VOS-895-P5NYK	Anti-Snoring Balloons	r_a7c4t	n_a7c4
16	EUK-638-CQ3LI	Self-Fluffing Backpack	r_a5c23t	n_a5c23
17	IFD-934-VVB3X	Anti-Gravity Chocolat...	r_a1c8t	n_a1c8


Resources

- GitHub Repo:
<https://github.com/SimioLLC/PythonExamplesInSimio>
- Simio Help
- Simio API Reference Guide
- My Account

Example Models – Simio GitHub Site

- <https://github.com/SimioLLC/PythonExamplesInSimio>


 01 Python Integration - Test Initial Python Configuration Model.docx

 02 Python Integration - General States and Properties.docx

 03 Python Integration - Data Tables with State Columns.docx


 04 Python Integration - Write to Output Table.docx


 05 Python Integration - Server Selection Model.docx


 06 Python Integration - Pandas DataFrames From Tables.docx


 07 Python Integration - Using ModelGlobalData.docx


 08 Python Integration - Testing Shared Files.docx


 Callback Test V01.spfx


 Data Tables With State Columns V01.spfx


 General States and Properties V01.spfx


 Pandas DataFrames from Tables V01.spfx

 Server Selection V01.spfx

 Test Initial Python Configuration V01.spfx

 Testing Shared Files V01.spfx

 Using ModelGlobalData V01.spfx

 Write to Output Table V01.spfx