

Sim2Ls Structure by Example

<https://nanohub.org/tools/introto simtools>

Dashboard

Introduction to SimTools

By Saaketh Desai, Steven Clark, Alejandro Strachan

Learn about SimTools - a product to deliver simulations with validated inputs, outputs and simulation caching in nanoHUB

Edit

Launch Tool

Version 1.21 - published on 04 Oct 2021

doi:10.21981/W91P-6R20 cite this

Open source: [license](#) | [download](#)

[View All Supporting Documents](#)

About Usage Citations Questions Reviews Wishlist Versions Supporting Docs Usage (New)

Category

Tools

Published on

04 Oct 2021

Abstract

This tool demonstrates SimTools, the latest way to deliver online simulations in nanoHUB. SimTools are Jupyter notebooks that include declarations of inputs and outputs and a simulation workflow to obtain the outputs from the inputs. The workflow can include physics-based simulations together with pre- and post-processing, or a simple function evaluation. SimTool developers declare inputs (including units and ranges) as well as outputs and the SimTool libraries validates inputs before executing the workflow. SimTool runs that execute correctly and result in valid outputs are automatically added to the nanoHUB simulation cache, so they do not need to be re-executed if the same run is subsequently requested.

nanoHUB users involve the SimTools from graphical user interface apps (see for example: (<https://nanohub.org/tools/qdotjuptest>) or from workflows (see <https://nanohub.org/tools/meltingkim>).

This tool showcases the mechanics of setting up a SimTool and an associated workflow, describing the variety of input and output types possible and the basics of setting up a Run and saving results in the nanoHUB cache. SimTools documentation can be found at: <https://simtool.readthedocs.io/en/latest/>

Search “simtool”

MY TOOLS

Recent

Favorites

All Tools

simtool

Introduction to SimTools

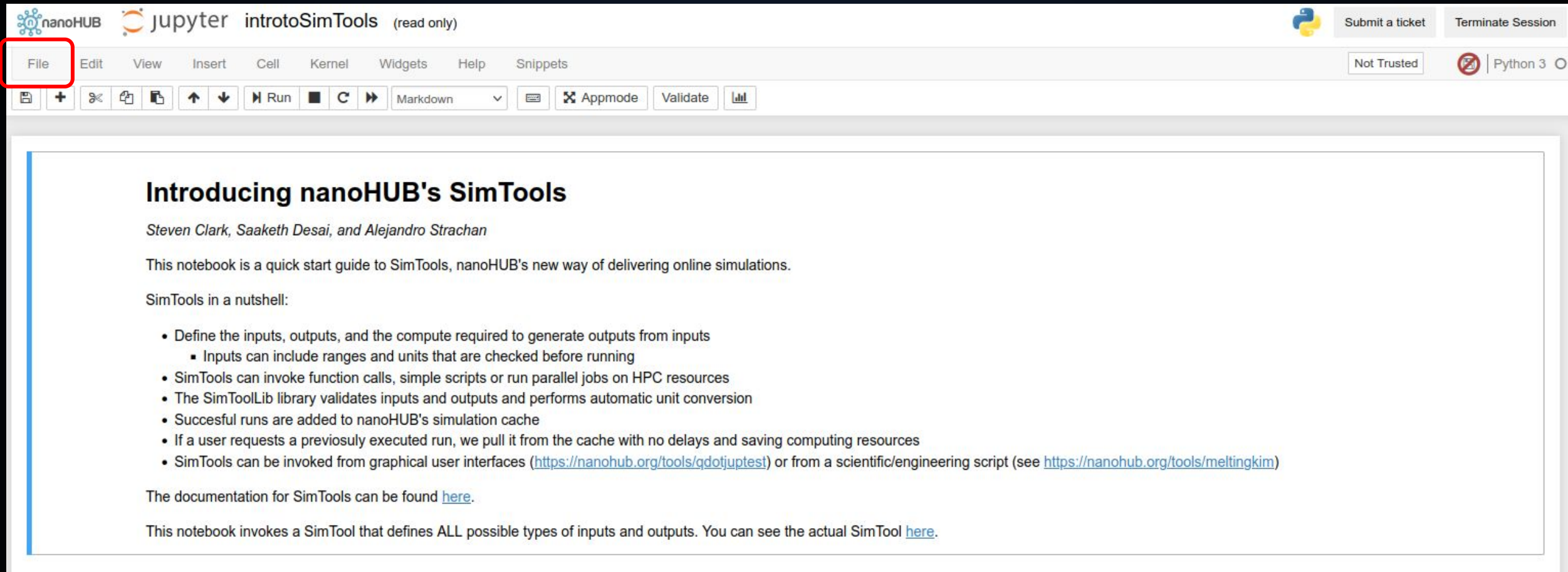


pntoy using simtool infrastructure



Add a tool to your favorites by clicking a heart. Click the heart again to remove it.

introSimTools - workflow notebook



The screenshot shows the nanoHUB Jupyter interface. The top bar includes the nanoHUB logo, the Jupyter logo, and the notebook title "introSimTools (read only)". On the right, there are links for "Submit a ticket" and "Terminate Session". Below the top bar is a menu bar with "File", "Edit", "View", "Insert", "Cell", "Kernel", "Widgets", "Help", and "Snippets". The "File" menu is highlighted with a red box. Below the menu bar is a toolbar with icons for saving, opening, and running files, as well as buttons for "Appmode", "Validate", and a chart icon. The main content area displays the notebook text.

Introducing nanoHUB's SimTools

Steven Clark, Saaketh Desai, and Alejandro Strachan

This notebook is a quick start guide to SimTools, nanoHUB's new way of delivering online simulations.



SimTools in a nutshell:

- Define the inputs, outputs, and the compute required to generate outputs from inputs
 - Inputs can include ranges and units that are checked before running
- SimTools can invoke function calls, simple scripts or run parallel jobs on HPC resources
- The SimToolLib library validates inputs and outputs and performs automatic unit conversion
- Successful runs are added to nanoHUB's simulation cache
- If a user requests a previously executed run, we pull it from the cache with no delays and saving computing resources
- SimTools can be invoked from graphical user interfaces (<https://nanohub.org/tools/qdotjuptest>) or from a scientific/engineering script (see <https://nanohub.org/tools/meltingkim>)

The documentation for SimTools can be found [here](#).

This notebook invokes a SimTool that defines ALL possible types of inputs and outputs. You can see the actual SimTool [here](#).

introto simtools - file explorer



 



Submit a ticket Terminate Session

Files Running Formgrader Assignments Courses

Select items to perform actions on them.

Upload New ↻

| <input type="checkbox"/> 0 ▾ | / | Name ▾ | Last Modified | File size |
|------------------------------|--|---------|---------------|-----------|
| <input type="checkbox"/> | bin | | 7 minutes ago | |
| <input type="checkbox"/> | data | | 7 minutes ago | |
| <input type="checkbox"/> | doc | | 7 minutes ago | |
| <input type="checkbox"/> | examples | | 7 minutes ago | |
| <input type="checkbox"/> | middleware | | 7 minutes ago | |
| <input type="checkbox"/> | simtool | | 7 minutes ago | |
| <input type="checkbox"/> | src | | 7 minutes ago | |
| <input type="checkbox"/> |  introtoSimTools.ipynb | Running | 7 minutes ago | 47 B |
| <input type="checkbox"/> |  introtosimtoolsExample.ipynb | | 7 minutes ago | 54 B |
| <input type="checkbox"/> | LICENSE.txt | | 7 minutes ago | 37 B |
| <input type="checkbox"/> | README.md | | 7 minutes ago | 35 B |


 

Submit a ticket Terminate Session

Files Running Formgrader Assignments Courses

Select items to perform actions on them.

Upload New ↻

| <input type="checkbox"/> 0 ▾ | / simtool | Name ▾ | Last Modified | File size |
|------------------------------|---|--------|----------------|-----------|
| <input type="checkbox"/> | .. | | seconds ago | |
| <input type="checkbox"/> |  introtosimtools.ipynb | | 10 minutes ago | 55 B |

introtoolsimtools - sim2L notebook

Introducing nanoHUB's SimTools

Steven Clark, Saaketh Desai, and Alejandro Strachan

This notebook is a quick start guide to SimTools, nanoHUB new way of delivering online simulations.

SimTools in a nutshell

- Define the inputs, outputs, and the compute required to generate outputs from inputs
 - Inputs can include ranges and units that are checked before running
- SimTools can invoke parallel jobs running on HPC resources, a simple script, or evaluating a function call
- The SimToolLib library validates inputs and outputs, including automatic unit conversion
- Successful runs are added to nanoHUB's simulation cache
- If a user requests a previously executed run, we pull it from the cache with no delays and saving computing resources
- SimTools can be invoked from graphical user interfaces (example here) or from a scientific/engineering script (example here)

The documentation for SimTools can be found [here](#).

This notebook demonstrates a simple SimTools that demonstrates all the possible input and output types.

Step 1. Provide a description of your tool

This cell is optional but highly recommended. The provided description is displayed when returning SimTool search results.

In []:

DESCRIPTION ✕

```
DESCRIPTION = """Show examples of SimTool input and output types"""
```

introtosimtools - sim2L notebook

In []:

```
%load_ext yamlmagic
```

Step 2. Define all inputs. Including valid ranges and units is strongly encouraged

In []:

```
%%yaml INPUTS
```

```
booleanValue:  
  type: Boolean  
  description: Execute bogus save operations  
  value: False
```

```
textString:  
  type: Text  
  description: Text supplied as string  
  value: textString
```

```
textFile:  
  type: Text  
  description: Text supplied as file
```

```
integerValue:  
  type: Integer  
  description: Simple integer  
  value: 10
```

```
numberValue:  
  type: Number  
  description: Simple number  
  value: 10.5  
  min: 0.0  
  max: 100.0  
  units: K
```

```
# Specifying units clearly defines the units that the SimTool expects for the input.  
# Examples of inputs are K (for temperature), angstroms (for distance) and fs for time.  
# If the input from the user is in some other units, we use the Pint library (https://pint.readthedocs.io/en/0.10.1/)  
# to automatically convert the units, and, in combination with range checking, validating inputs for all SimTool runs.  
# See the pint documentation for a list of allowable units.
```

introtoolsimtools - sim2L notebook

In []:

FILES ✕

```
# If you simulation require additional files (e.g. configuration files), list them here. The files  
# should exist in the same directory (simtool) as this notebook.  
# This cell is optional. The tag FILES and variable EXTRA_FILES must specified exactly as given here.  
EXTRA_FILES = []
```

In []:

parameters ✕

```
from simtool import getValidatedInputs  
  
defaultInputs = getValidatedInputs(INPUTS)  
if defaultInputs:  
    globals().update(defaultInputs)
```

introtosimtools - sim2L notebook

In []:

```
%%yaml OUTPUTS
```

```
booleanValue:
```

```
  type: Boolean
```

```
  description: Execute bogus save operations
```

```
  value: False
```

```
textString:
```

```
  type: Text
```

```
  description: Text supplied as string
```

```
  value: textString
```

```
textFile:
```

```
  type: Text
```

```
  description: Text supplied as file
```

```
integerValue:
```

```
  type: Integer
```

```
  description: Simple integer
```

```
  value: 10
```

```
numberValue:
```

```
  type: Number
```

```
  description: Simple number
```

```
  value: 10.5
```

```
  min: 0.
```

```
  max: 100.
```

```
arrayValue:
```

```
  type: Array
```

```
  description: Array of numbers
```

```
  value: [1,2,3]
```


intro simtools - sim2L notebook

In []:

```
db = DB(OUTPUTS)
```

In []:

```
print(booleanValue)
db.save('booleanValue', booleanValue)

print(integerValue)
db.save('integerValue', integerValue)

print(numberValue)
db.save('numberValue', numberValue)

print(textString)
db.save('textString', textString)

print(textFile)
copyAndSaveFileAsOutput('textFile', textFile)

print(imageFile)
copyAndSaveFileAsOutput('imageFile', imageFile)

print(imageValue)
db.save('imageValue', imageValue)

print(arrayValue)
db.save('arrayValue', arrayValue)

print(arrayFile)
copyAndSaveFileAsOutput('arrayFile', arrayFile)

print(listValue)
db.save('listValue', listValue)

print(listFile)
copyAndSaveFileAsOutput('listFile', listFile)

print(dictValue)
db.save('dictValue', dictValue)
```


introtools - sim2L notebook

```
print(dictValue)
db.save('dictValue', dictValue)

print(dictFile)
copyAndSaveFileAsOutput('dictFile', dictFile)

print(choiceValue)
db.save('choiceValue', choiceValue)

print(elementValue)
db.save('elementValue', elementValue)
```

False

10

10.5

textString

None

None

None

[1, 2, 3]

None

['one', 'two', 'three']

None

{'one': 1, 'two': 2, 'three': 3}

None

pear

58.6934

introtoolsimtools - workflow notebook

Step 1. Setting things up

```
In [1]: # We will import various libraries including key elements of nanoHUB's simtool library
        %load_ext yamlmagic

        import os
        import numpy as np
        import PIL.Image

        from simtool import findInstalledSimToolNotebooks, searchForSimTool
        from simtool import getSimToolInputs, getSimToolOutputs, Run

In [2]: # Identify the simtool of interest (in this case introducing simtools) and retrieve its status
        simToolName = "introtoolsimtools"
        simToolLocation = searchForSimTool('introtoolsimtools')
        for key in simToolLocation.keys():
            print("%18s = %s" % (key, simToolLocation[key]))

        notebookPath = /apps/introtoolsimtools/r13/simtool/introtoolsimtools.ipynb
        simToolName = introtoolsimtools
        simToolRevision = r13
        published = True
```

introtosimtools - workflow notebook

Step 2. Inputs

```
In [3]: # get the list of inputs for the simtool. This is an exhaustive list of inputs for SimTools.  
inputs = getSimToolInputs(simToolLocation)  
inputs
```

```
integerValue:  
  min: None  
  max: None  
  type: Integer  
  description: Simple integer  
  value: 10
```

```
numberValue:  
  units: kelvin  
  min: 0.0  
  max: 100.0  
  type: Number  
  description: Simple number  
  value: 10.5
```

```
arrayValue:  
  min: None  
  max: None  
  type: Array
```

introto simtools - workflow notebook

```
In [4]: inputs['booleanValue'].value = False
inputs['integerValue'].value = 5
inputs['numberValue'].value = 10.7
inputs['textString'].value = "Now is the time for all good men to come to the aid of their party"
inputs['textFile'].file = os.path.join("data", "Text", "party.txt")
inputs['imageFile'].file = os.path.join("data", "Images", "dome_qd_simple.png")
inputs['imageValue'].value = PIL.Image.open(os.path.join("data", "Images", "jup.png"))
inputs['arrayValue'].value = np.array([[1., 2., 3.], [4., 5., 6.], [7., 8., 9.]])
inputs['arrayFile'].file = os.path.join("data", "Array", "2Darray.json")
inputs['listValue'].value = ['one', 'two', 'three', 'seven']
inputs['listFile'].file = os.path.join("data", "List", "list.json")
inputs['dictValue'].value = {'one': 2, 'two': 2, 'three': 3}
inputs['dictFile'].file = os.path.join("data", "Dict", "dict.json")
inputs['choiceValue'].value = "apple"
inputs['elementValue'].value = "Fe"

inputs
```

```
integerValue:
  min: None
  max: None
  type: Integer
  description: Simple integer
  value: 5
```

```
numberValue:
  units: kelvin
  min: 0.0
  max: 100.0
  type: Number
  description: Simple number
  value: 10.7
```

```
arrayValue:
  min: None
  max: None
  type: Array
```


introSimtools - workflow notebook

```
In [5]: # We can explore the outputs the SimTool will produce before running the simulation.  
# Of course, at this point all output variables are empty  
outputs = getSimToolOutputs(simToolLocation)  
outputs
```

```
Out[5]: booleanValue:  
    type: Boolean  
    description: Execute bogus save operations  
    value: False  
  
    textString:  
        type: Text  
        description: Text supplied as string  
        value: textString  
  
    textFile:  
        type: Text  
        description: Text supplied as file  
  
    integerValue:  
        min: None  
        max: None  
        type: Integer  
        description: Simple integer  
        value: 10
```

introtosimtools - workflow notebook

Step 3. Run the SimTool

In [6]: `r = Run(simToolLocation,inputs)`

```
submit --local /apps/bin/ionhelperGetArchivedSimToolResult.sh introtosimtools
      r13 RUNS/447c77ed05f446488413f27206d6fcea/inputs.yaml
      RUNS/447c77ed05f446488413f27206d6fcea
Found cached result
```

introtoolsimtools - workflow notebook

Step 4. Get the outputs

In [7]: `r.getResultSummary()`

Out[7]:

| | name | data | encoder | display | filename |
|----|--------------------------|--|---------|---------|-------------------------|
| 0 | simToolSaveErrorOccurred | 0 | text | None | introtoolsimtools.ipynb |
| 1 | simToolAllOutputsSaved | 1 | text | None | introtoolsimtools.ipynb |
| 2 | booleanValue | false | text | None | introtoolsimtools.ipynb |
| 3 | integerValue | 5 | text | None | introtoolsimtools.ipynb |
| 4 | numberValue | 10.7 | text | None | introtoolsimtools.ipynb |
| 5 | textString | "Now is the time for all good men to come to t..." | text | None | introtoolsimtools.ipynb |
| 6 | textFile | file://party.txt | text | None | introtoolsimtools.ipynb |
| 7 | imageFile | file://dome_qd_simple.png | text | None | introtoolsimtools.ipynb |
| 8 | imageValue | [[[255, 255, 255], [255, 255, 255], [255, 255, ... | text | None | introtoolsimtools.ipynb |
| 9 | arrayValue | [[1.0, 2.0, 3.0], [4.0, 5.0, 6.0], [7.0, 8.0, ... | text | None | introtoolsimtools.ipynb |
| 10 | arrayFile | file://2Darray.json | text | None | introtoolsimtools.ipynb |
| 11 | listValue | ["one", "two", "three", "seven"] | text | None | introtoolsimtools.ipynb |
| 12 | listFile | file://list.json | text | None | introtoolsimtools.ipynb |
| 13 | dictValue | {"one": 2, "three": 3, "two": 2} | text | None | introtoolsimtools.ipynb |
| 14 | dictFile | file://dict.json | text | None | introtoolsimtools.ipynb |
| 15 | choiceValue | "apple" | text | None | introtoolsimtools.ipynb |
| 16 | elementValue | 55.845 | text | None | introtoolsimtools.ipynb |

introtoolsimtools - workflow notebook

```
In [8]: resultBooleanValue = r.read('booleanValue')  
        print(resultBooleanValue)
```

False

```
In [9]: resultIntegerValue = r.read('integerValue')  
        print(resultIntegerValue)
```

5

```
In [10]: resultNumberValue = r.read('numberValue')  
         print(resultNumberValue)
```

10.7