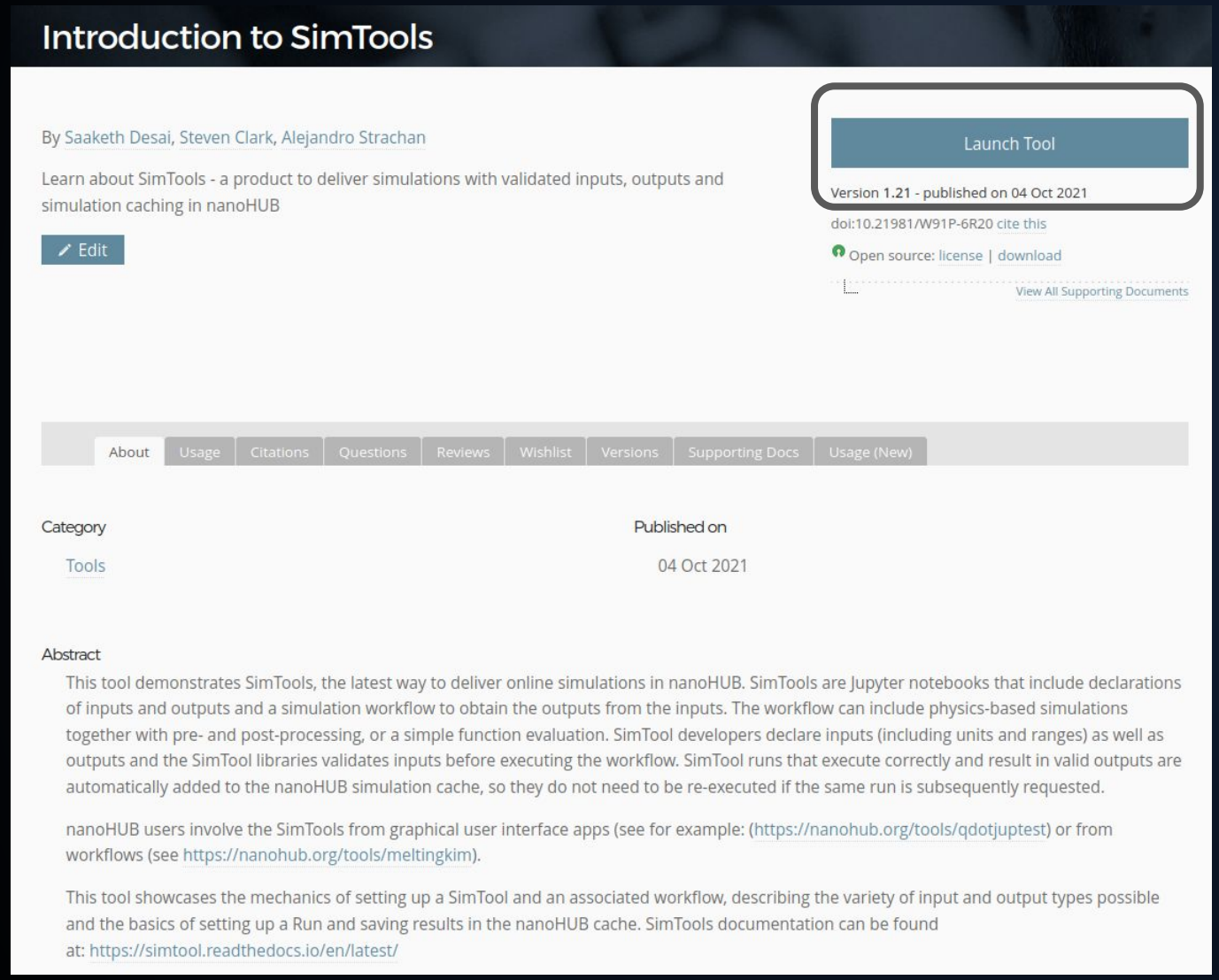
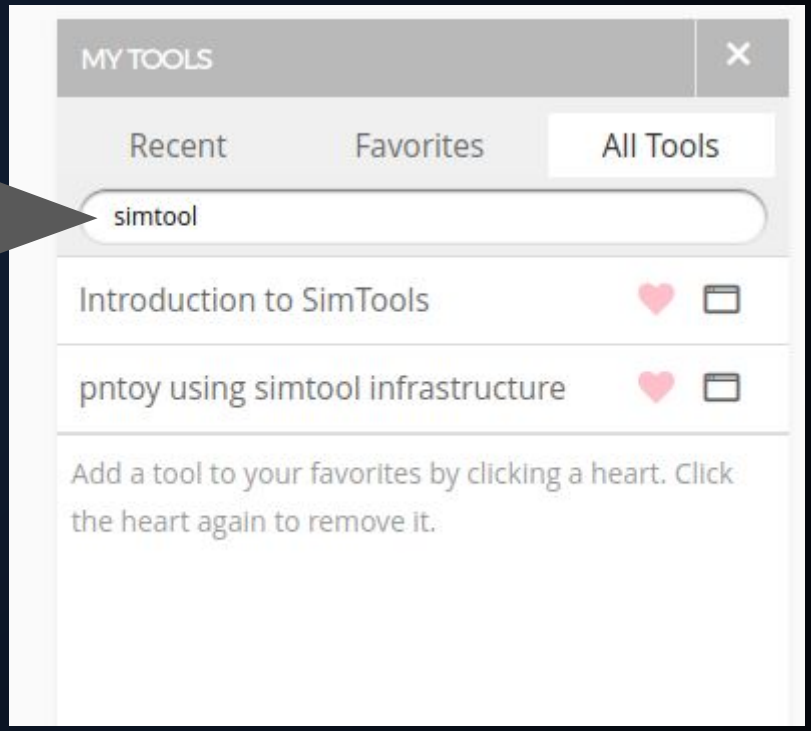
Sim2Ls Structure by Example

https://nanohub.org/tools/introtosimtools

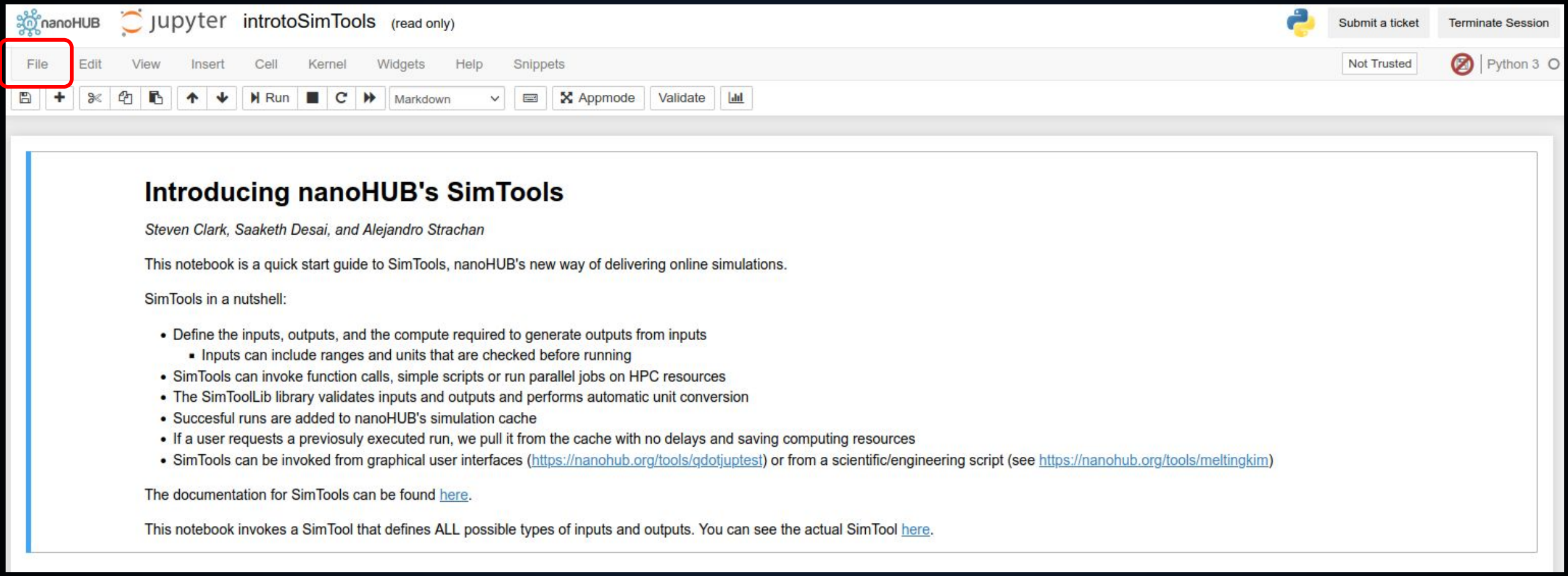


Dashboard

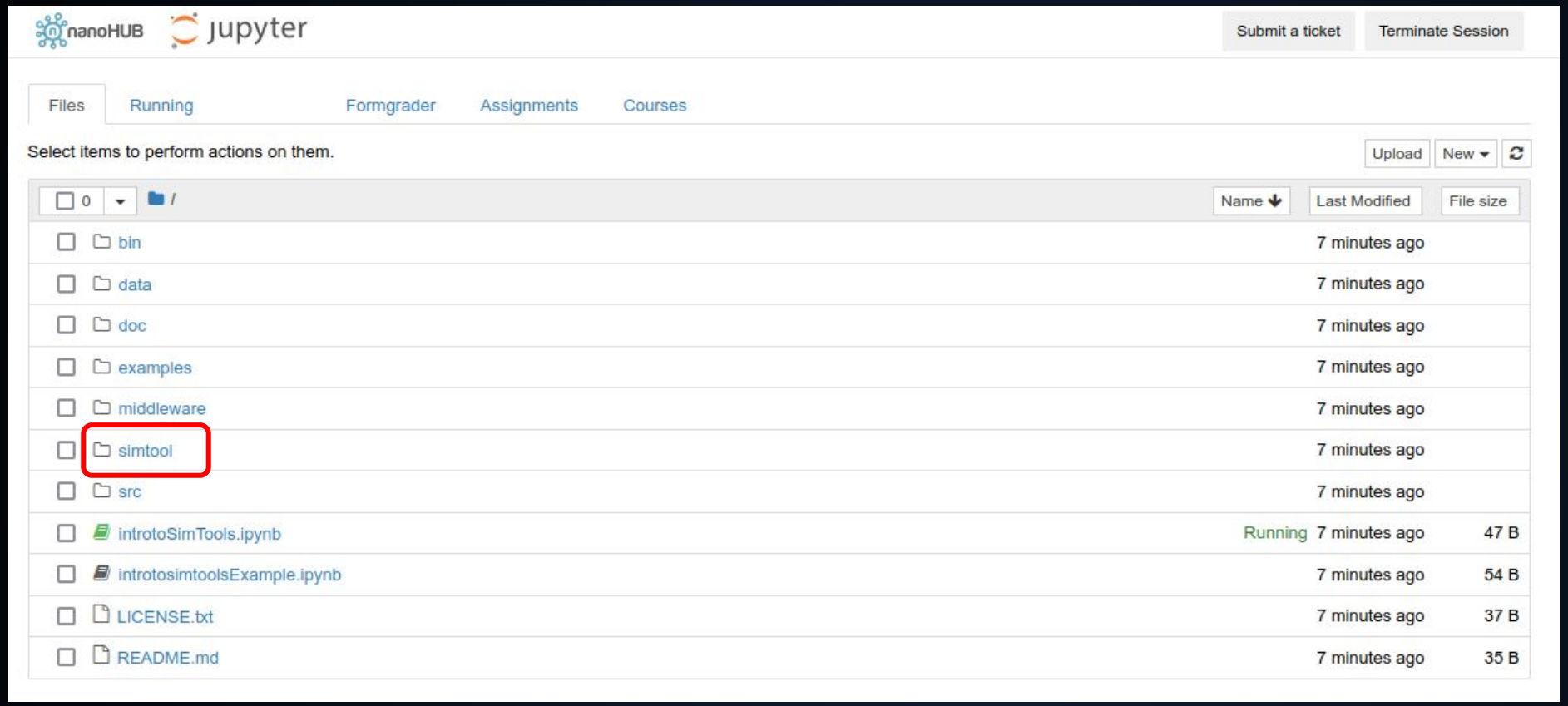
Search “simtool”

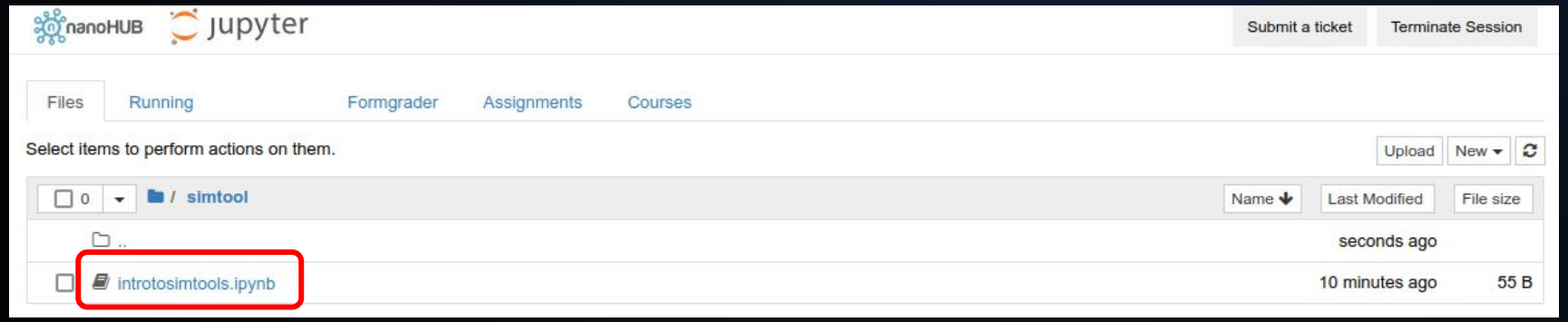


introtosimtools - workflow notebook



introtosimtools - file explorer





introtosimtools - 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
- Succesful runs are added to nanoHUB's simulation cache
- If a user requests a previosuly 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.
```

introtosimtools - 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]
```

introtosimtools - 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)
```

introtosimtools - 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
```

introtosimtools - 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 introducingsimtools) and retrieve its status
simToolName = "introtosimtools"
simToolLocation = searchForSimTool('introtosimtools')
for key in simToolLocation.keys():
    print("%18s = %s" % (key,simToolLocation[key]))
```

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

introtosimtools - workflow notebook

Step 2. Inputs

In [3]:
```
# get the list of inputs for the simtool. This is an exhausitive 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
```

introtosimtools - 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
```

introtosimtools - 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
```

introtosimtools - workflow notebook

Step 4. Get the outputs

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

Out[7]:

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

introtosimtools - 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
```