Simulation using Jupyter Notebook

You can simulate Teamwork Cloud projects on the server by using Jupyter Notebook. This chapter explains how to set up Jupyter Notebook for server-side simulation and lists all available commands with examples.

Prerequisites

Before starting the simulation, make sure you have prepared your projects for server-side simulation.

Setting up Jupyter Notebook

To set up Jupyter Notebook

- In Jupyter Notebook, run the following command to install a Python package from the *<install_root>\plugins\com.nomagic.magicdraw.simulation\pyST.zip* file:

```
%pip install pyST.zip
```

Use the following requests to simulate Teamwork Cloud projects on the server:

Create client/session and authenticate

```
client = SimulationWebClient('http(s)://<server_host>:<server_port>', '<TWC_user_name>', '<TWC_user_password>', False)
```

This request starts a work session and provides authentication to Teamwork Cloud.

The following table describes the parameters used in the request:

Parameter	In	Required or optional	Description
server_host	path	required	The Teamwork Cloud server host name.
server_port	path	required	The Teamwork Cloud server port.
TWC_user_name	path	optional	The Teamwork Cloud user name. If you do not specify the user name in the request, an input field will be provided to specify it later.
TWC_user_password	path	optional	The Teamwork Cloud password. If you do not specify the password in the request, an input field will be provided to specify it later.

For instructions on how to run server-side simulation using SSL, refer to Running server-side simulation with SSL

Get the list of Teamwork Cloud projects

```
client.get_projects()
```

This request provides the list of Teamwork Cloud projects accessible to you depending on your permissions. Only projects that you can read (the Read Resources permission) are returned.

Request and response examples

```
client.get_projects()


[{'id': '1f9fd988-620b-4b3c-8414-7ab96dc53a48',
  'name': 'SpacecraftMassRollup',
  'description': ''},
 {'id': 'c125b91e-7d31-4015-b659-6fd98059b8c7',
  'name': 'BouncingBall',
  'description': 'This project demonstrates the execution of the BouncingBall, including fmu.'},
 {'id': '485dfc82-7b08-43cd-86b1-953b3725e5b1',
  'name': 'CarBrakingAnalysis',
  'description': 'This project calculates the stopping distance based on car speed and mass.'}]
```

Run simulation

client.run(<project>, version=<version>, branch=<branch>, element_id=<element_id>, config=<config>, commit_results=<True/False>, verification=<All/None/Fail>, data=json.dumps(<param>), auto_start=<True/False>)

This request initiates the execution of the specified Simulation Configuration in a particular project. It connects to Teamwork Cloud, finds the element to execute, and initiates the execution if the element exists. After the execution is complete, the request constructs and returns a unique ID (per application) for the given execution.

The following table describes the parameters used in the request:

Parameter	In	Required or optional	Description
project	path	required	The Teamwork Cloud project name or ID.
version	path	optional	The Teamwork Cloud project version.
branch	path	optional	The project branch name or ID. If the branch is omitted, the trunk is used instead.
element_id	path	optional	The server ID of the Instance Specification to be executed.
config	path	required	The Simulation Configuration name or server ID.
auto_start	path	optional	Starts the simulation automatically after the initialization phase is completed. Available values are: ○ *true* (default) - performs the initialization phase and then starts the simulation (the simulation clock is started). ○ *false* - only the initialization phase of the simulation is performed. The simulation then waits for an invocation to start the simulation, do a step, or terminate the simulation. ⚠ The **?autoStart** query parameter is ignored when multiple iterations are running, e.g. Tradestudy, Montecarlo, or table simulation
started_from	path	optional	Allows specifying the starting location of the simulation. The value of the **started_from** parameter is provided in the status endpoint response.
commit_results	path	optional	Commits a new project version with the simulation results. Available values are *True* or *False* (default).

verification	path	optional	Returns the selected verification results. Available values are: ○ **All** - all verification results are returned. ○ **None** - no verification results are returned. ○ **Fail** (default) - verification results with the fail status are returned.
data	path	optional	A set of output parameters, which will be obtained after the simulation is complete.

Request and response examples

```
 # SpaceCraftMassRollup sample

parameters = {
    "inputs":
   {
       "telecom.antenna.me":10,
        "telecom.amplifier.me":20
   },
 "outputs":
    [
        "me",
        "propulsion.me",
        "propulsion.tank.me",
        "propulsion.thruster.me",
        "telecom.me",
        "telecom.antenna.me",
        "telecom.amplifier.me"
    ]
}

client.run('SpacecraftMassRollup_SimWeb', config='spacecraft mass analysis', commit_results=False, data=json.
dumps(parameters))
```

Perform a time step

client.step(<simulation_id>)

This request performs a single time step of the specified simulation. To run the simulation by step, it should be run with the **auto_start** query parameter set to *false* .

The following table describes the parameters used in the request:

Parameter	In	Required or optional	Description
simulation_id	path	required	The ID of a specific simulation.

Start simulation

client.start(<simulation_id>)

This request starts the execution of the specified simulation. If the initialization phase is still in progress, the start endpoint is memorized and sent when the initialization phase is completed.

The following table describes the parameters used in the request:

Parameter	In	Required or optional	Description
simulation_id	path	required	The ID of a specific simulation.

Start simulation and get results

client.simulate(<project>, version=<version>, branch=<branch>, element_id=<element_id>, config=<config>, commit_results=<True/False>, verification=<All/None/Fail>, data=json.dumps(<param>))

This request starts simulation and returns its results.

The following table describes the parameters used in the request:

Parameter	In	Required or optional	Description
project	path	required	The Teamwork Cloud project name or ID.
version	path	optional	The Teamwork Cloud project version.
branch	path	optional	The project branch name or ID. If the branch is omitted, the trunk is used instead.
element_id	path	optional	The server ID of the Instance Specification to be executed.
config	path	required	The Simulation Configuration name or server ID.
commit_results	path	optional	A new project version is committed with the simulation results. Available values are *True* or *False* (default).
verification	path	optional	Returns the selected verification results. Available values are: ○ **All** - all verification results are returned. ○ **None** - no verification results are returned. ○ **Fail** - verification results with the fail status are returned including the main status and the status of requirements and constraints. If the **verification** value is not specified, only the verification results with the fail status are returned when the get_results endpoint is called.
data	path	optional	A set of output parameters, which will be obtained after the simulation is complete.

Request and response examples

```
 # SpaceCraftMassRollup sample

parameters = {
    "inputs":
   {
       "telecom.antenna.me":10,
        "telecom.amplifier.me":20
   },
 "outputs":
    [
        "me",
        "propulsion.me",
        "propulsion.tank.me",
        "propulsion.thruster.me",
        "telecom.me",
        "telecom.antenna.me",
        "telecom.amplifier.me"
    ]
}

client.simulate('SpacecraftMassRollup_SimWeb', config='spacecraft mass analysis', commit_results=False,
data=json.dumps(parameters))
```

Get simulation status

client.get_status(<simulation_id>)

This request gets the status of a specific simulation.

The following table describes the parameters used in the request:

Parameter	In	Required or optional	Description
simulation_id	path	required	The ID of a specific simulation.

Request and response examples

```
client.get_status('55089221-8673-4391-aa3e-a8e774ecfa48')

{'state': 'RUNNING',
 'simulationId': '55089221-8673-4391-aa3e-a8e774ecfa48',
 'simulationTime': '0 ms',
 'ui': '/simulation/api/ui/55089221-8673-4391-aa3e-a8e774ecfa48/CoffeeMachine.html',
 'project': 'CoffeeMachine',
 'config': 'Coffee Machine Web',
 'elapsedTime': 6598}


client.get_status('ce8c8215-0515-43fd-9d34-92d1d7a95d87')

{'state': 'COMPLETED',
 'simulationId': 'ce8c8215-0515-43fd-9d34-92d1d7a95d87',
 'simulationTime': '3000 ms',
 'project': 'SpacecraftMassRollup',
 'config': 'spacecraft mass analysis',
 'elapsedTime': 6598}


client.get_status('b7bdf933-f58d-4e7e-b73b-8370c60485cd')

{'state': 'QUEUED',
 'queueNumber': 3,
 'simulationId': 'b7bdf933-f58d-4e7e-b73b-8370c60485cd',
 'simulationTime': '0 ms',
 'project': 'CarBrakingAnalysis',
 'elapsedTime': 52}
```

Get simulation variables

client.get_variables(<simulation_id>, variables=json.dumps(<parameters>))

This request returns a list of simulation variables during simulation.

The following table describes the parameters used in the request:

Parameter	In	Required or optional	Description
simulation_id	path	required	The ID of the running simulation.
variables	path	optional	The set of simulation variables that should be obtained. If no variables are specified, the values of all simulation variables are returned.

Set simulation variables

client.set_variables(<simulation_id>, variables=json.dumps(<parameters>))

This request provides a list of simulation variables with values.

The following table describes the parameters used in the request:

Parameter	In	Required or optional	Description
simulation_id	path	required	The ID of the running simulation.
variables	path	required	The set of simulation variables with values that should be used during the current simulation.
context	path	optional	The context ID.
property_path	path	optional	The property path of the context specified using property names or IDs.

Request and response examples

```
var = {
 "variables":
 {
 "position": 40
 }
}

client.set_variables('23ceff24-28fa-47e4-b29f-b6b60b4b12e3', variables=json.dumps(var))
```

Pause simulation

client.pause(<simulation_id>)

This request pauses the execution of the specified simulation.

The following table describes the parameters used in the REST API request:

Parameter	In	Required or optional	Description
simulation_id	path	required	The ID of the running simulation.

Resume simulation

client.resume(<simulation_id>)

This request resumes the execution of the specified simulation.

The following table describes the parameters used in the REST API request:

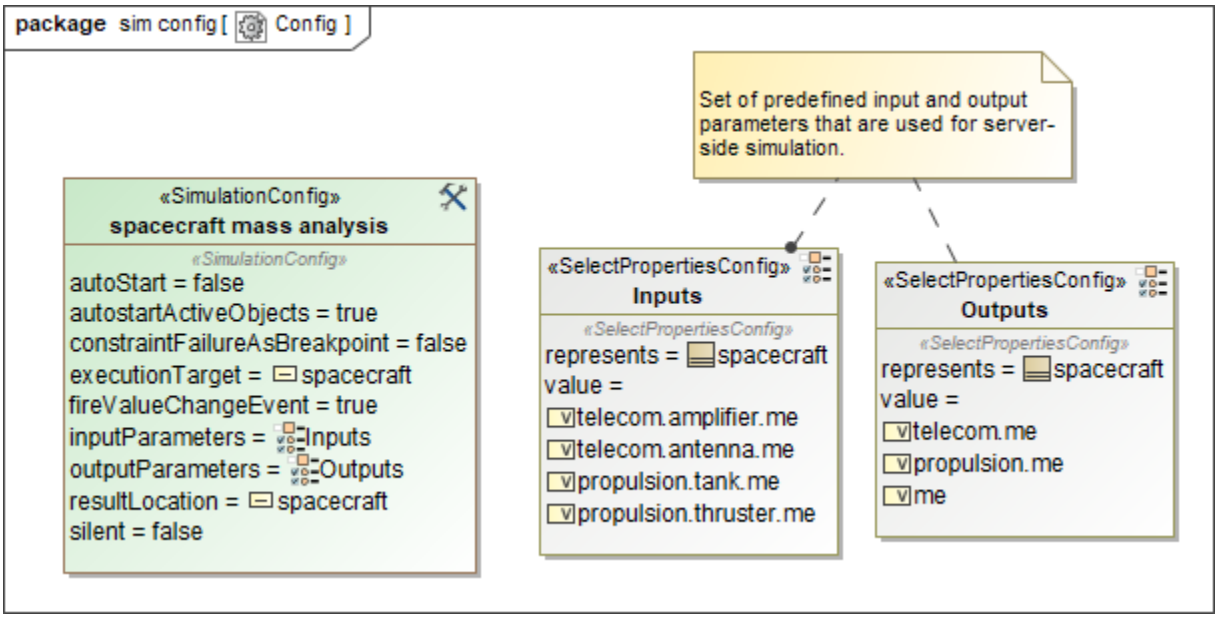
Parameter	In	Required or optional	Description
simulation_id	path	required	The ID of the running simulation.

Get simulation results

client.get_result(<simulation_id>)

This request returns the results of the specified simulation. If the **OutputParameters** tag of the executed Simulation Configuration is specified, the results are provided according to the **OutputParameters** tag value. If a Time Series or Timeline chart is specified for the Simulation Configuration, the chart data is returned in the JSON format.

The figure below demonstrates how to specify input and output parameters.



Specifying input and output parameters.

The following table describes the parameters used in the request:

Parameter	In	Required or optional	Description
simulation_id	path	required	The ID of the running simulation.

Get the list of running simulations

```
client.get_running()
```

This REST API request gets the list of all currently running simulations including the queued simulations that are placed in a waiting line.

Terminate simulation

client.terminate(<simulation_id>)

This request terminates the specified simulation.

The following table describes the parameters used in the request:

Parameter	In	Required or optional	Description
simulation_id	path	required	The ID of a specific simulation.

Check for Simulation Configurations

client.has_configurations(<project>, version=<version>, branch=<branch>)

This request checks if the project has any Simulation Configurations.

The following table describes the parameters used in the request:

Parameter	In	Required or optional	Description
project	path	required	The Teamwork Cloud project name or ID.
version	path	optional	The Teamwork Cloud project version.
branch	path	optional	The project branch name or ID. If the branch is omitted, the trunk is used instead.

Get Simulation Configurations

```
client.get_configurations(<project>, version=<version>, branch=<branch>, element_id=<element_id>)
```

This request retrieves the names and descriptions of the Simulation Configurations available for the given project.

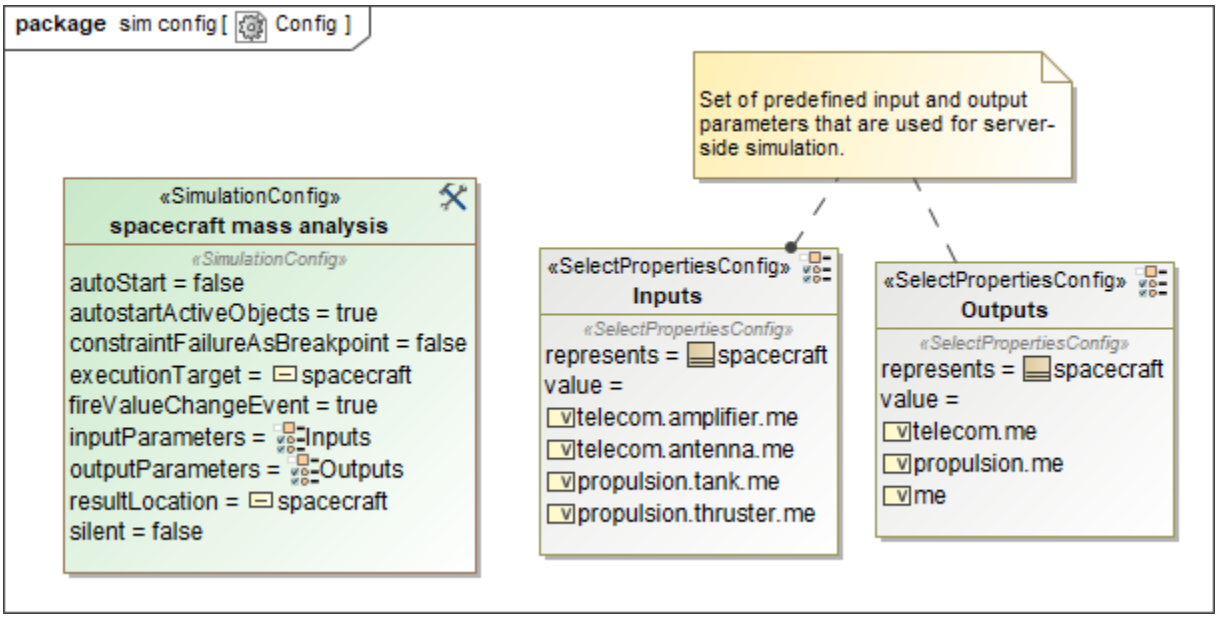
The following table describes the parameters used in the request:

Parameter	In	Required or optional	Description
project	path	required	The Teamwork Cloud project name or ID.
version	path	optional	The Teamwork Cloud project version.
branch	path	optional	The project branch name or ID. If the branch is omitted, the trunk is used instead.
element_id	path	optional	The server ID of the Instance Specification to be executed.
filter	path	optional	Returns Simulations Configurations with a specified UI tag for a particular project. The only possible value for the **filter** parameter is **ui**. Only Simulations Configurations having at least one of the following UI elements are returned: • Frame • Table • TimeSeries chart

Get Simulation Configuration descriptor

```
client.get_descriptor(<project>, version=<version>, branch=<branch>, config=<config>)
```

This REST API request retrieves Simulation Configuration data (description, execution target, time step, input and output parameters) from the specified project. The set of input and output parameters is specified using the **Input Parameters** and **Output Parameters** properties of a Simulation Configuration. The figure below demonstrates how to specify input and output parameters.



Specifying input and output parameters.

The following table describes the parameters used in the request:

Parameter	In	Required or optional	Description
project	path	required	The Teamwork Cloud project name or ID.
version	path	optional	The Teamwork Cloud project version.
branch	path	optional	The project branch name or ID. If the branch is omitted, the trunk is used instead.
config	path	required	The Simulation Configuration name or server ID.

Request and response examples

```
 client.get_descriptor('SpacecraftMassPollup_SimWeb', config='spacecraft mass analysis')

{'config': 'spacecraft mass analysis',
 'description': 'Simulation Config that is dedicated to run spacecraft mass rollup.',
 'model': 'spacecraft',
 'parameters': {'inputs': [{'parameter': 'telecom.amplifier.me',
    'value': 32.0,
    'type': 'mass[kilogram]',
    'unit': 'kilogram',
    'requirements': [{'id': '1',
      'text': 'Estimated mass shall be less than allocated mass'}]},
   {'parameter': 'telecom.antenna.me',
    'value': 32.0,
    'type': 'mass[kilogram]',
    'unit': 'kilogram',
    'requirements': [{'id': '1',
      'text': 'Estimated mass shall be less than allocated mass'}]},
   {'parameter': 'propulsion.tank.me',
    'value': 68.0,
    'type': 'mass[kilogram]',
    'unit': 'kilogram',
    'requirements': [{'id': '1',
      'text': 'Estimated mass shall be less than allocated mass'}]},
   {'parameter': 'propulsion.thruster.me',
    'value': 68.0,
    'type': 'mass[kilogram]',
    'unit': 'kilogram',
    'requirements': [{'id': '1',
      'text': 'Estimated mass shall be less than allocated mass'}]}],
  'outputs': [{'parameter': 'telecom.me',
    'value': 32.0,
    'type': 'mass[kilogram]',
    'unit': 'kilogram',
    'requirements': [{'id': '1',
      'text': 'Estimated mass shall be less than allocated mass'}]},
   {'parameter': 'propulsion.me',
    'value': 68.0,
    'type': 'mass[kilogram]',
    'unit': 'kilogram',
    'requirements': [{'id': '1',
      'text': 'Estimated mass shall be less than allocated mass'}]},
   {'parameter': 'me',
    'value': 95.0,
    'type': 'mass[kilogram]',
    'unit': 'kilogram',
    'requirements': [{'id': '1',
      'text': 'Estimated mass shall be less than allocated mass'}]}]}}


client.get_descriptor('BouncingBall', config='Run BouncingBall')


{'config': 'Run BouncingBall',
 'description': 'Simulation Config dedicated to run simulation with bouncingBall.fmu.',
 'model': 'bouncingBall',
 'timeStep': '10 ms'
 }
```