expecco - Quick Start Guide

Table of Contents

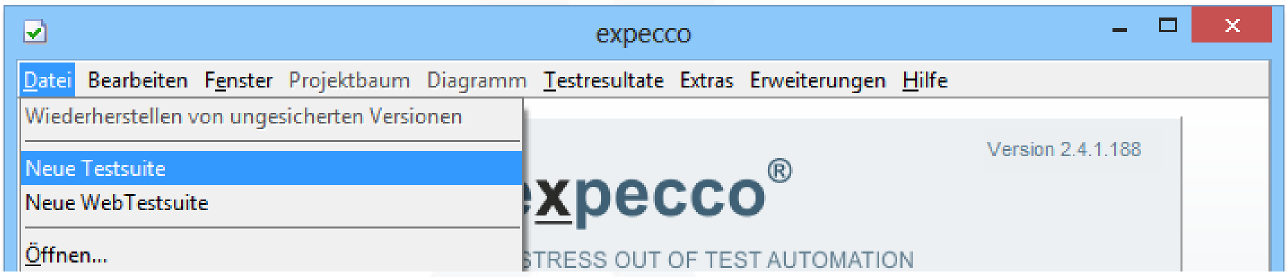
Introduction

The Quick Start Guide is aimed at everyone who wants to use **expecco** for the first time and will be a support to the user in the beginning.
We want to provide you an overview of **expecco**, using a simple example.
It is recommended to also start the demo tutorial in parallel in order to understand and follow everything.

Creating a new testsuite

A testsuite is the term used for a packaged test project. It manages a collection of elements like testplans, blocks and datatypes as well as resources or attachments and documentation.

To create a new testsuite, click on the menu *File → New Test Suite*.

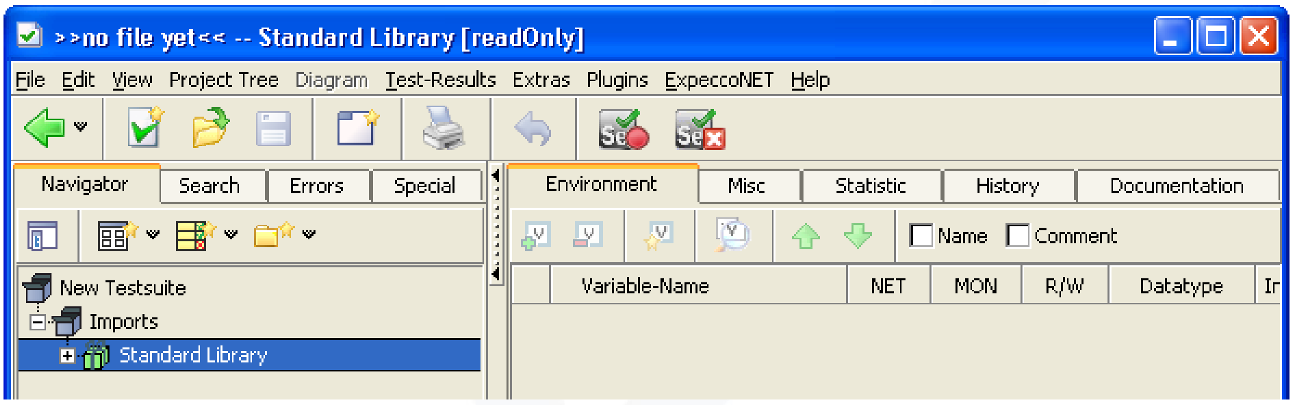


The Standard Library is automatically imported after creation of the new testsuite .

Blocks and libraries

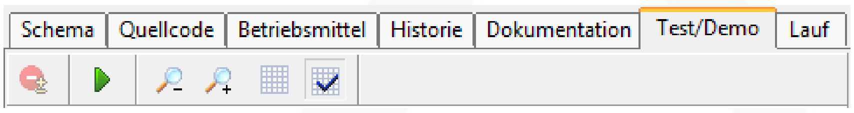
Libraries are collections of functional blocks. Libraries are located in the project tree (Navigator) under the category "Imports". All elements within an imported library are read-only. For each block, there is a description and, where appropriate, an example that you can execute immediately.

Select the Standard Library in the project tree. This library provides basic blocks for different areas.

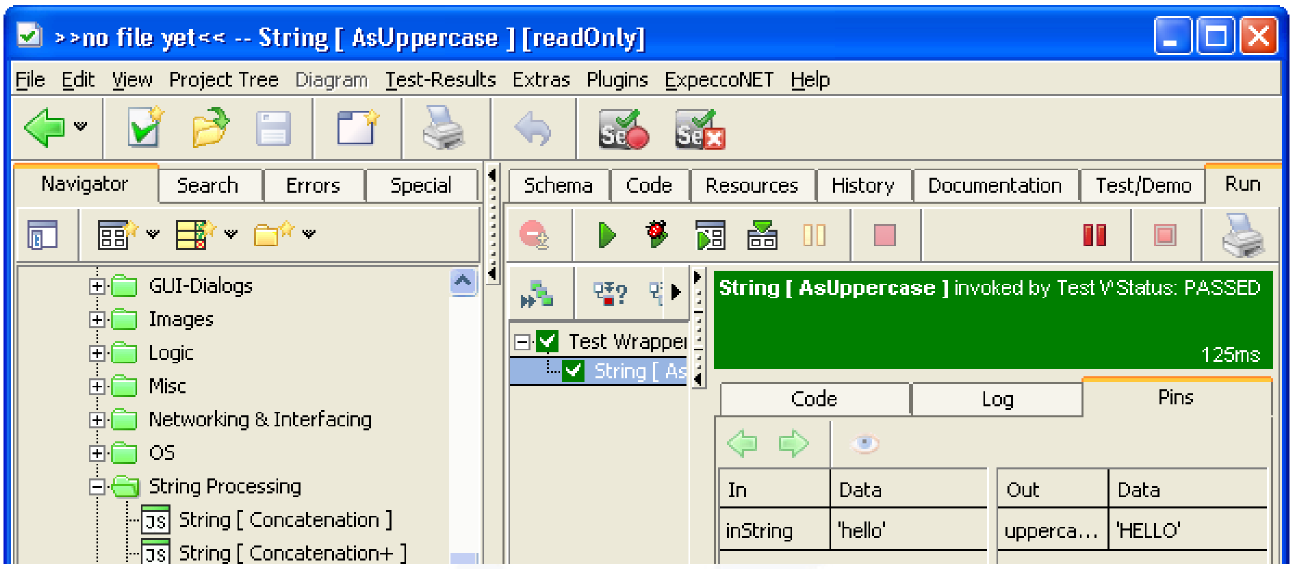

>>no file yet<< -- Standard Library [readOnly]
File Edit View Project Tree Diagram Test-Results Extras Plugins ExpeccoNET Help
Navigator Search Errors Special
Environment Misc Statistic History Documentation
Name Comment
New Testsuite
Imports
Standard Library
Variable-Name NET MON R/W Datatype

Within a library its blocks are organized in categories within folders. All blocks for string processing for example can be found in the same folder. Select in the category "String Processing" the sub folder "Padding & Formatting" and then select the block "String [AsUppercase]".

Editors regarding the selected item are displayed on the right in a number of tabs. In the tab "Documentation" you will find a brief description about the functionality.


Schema Quellcode Betriebsmittel Historie Dokumentation Test/Demo Lauf

Select the tab "Test". The shown Testeditor allows the immediate execution of the block by pressing the button .


>>no file yet<< -- String [AsUppercase] [readOnly]
File Edit View Project Tree Diagram Test-Results Extras Plugins ExpeccoNET Help
Navigator Search Errors Special
Schema Code Resources History Documentation Test/Demo Run
GUI-Dialogs
Images
Logic
Misc
Networking & Interfacing
OS
String Processing
String [Concatenation]
String [Concatenation+]
Test Wrapper
String [As
String [AsUppercase] invoked by Test V Status: PASSED
125ms
Code Log Pins

In	Data	Out	Data
inString	'hello'	upperca...	'HELLO'

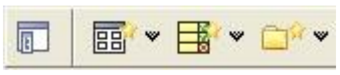
The result of the execution is displayed in the lower part. In the tab "Pins" you can see the input and output values.

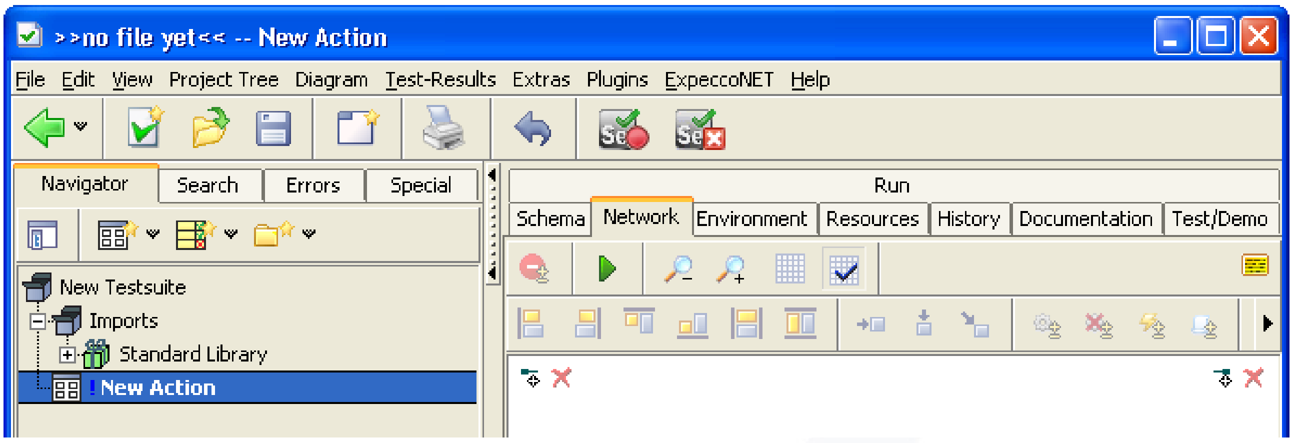
Creating your first (compound) block

A block is the definition of an activity. We distinguish between elementary blocks and compound blocks.

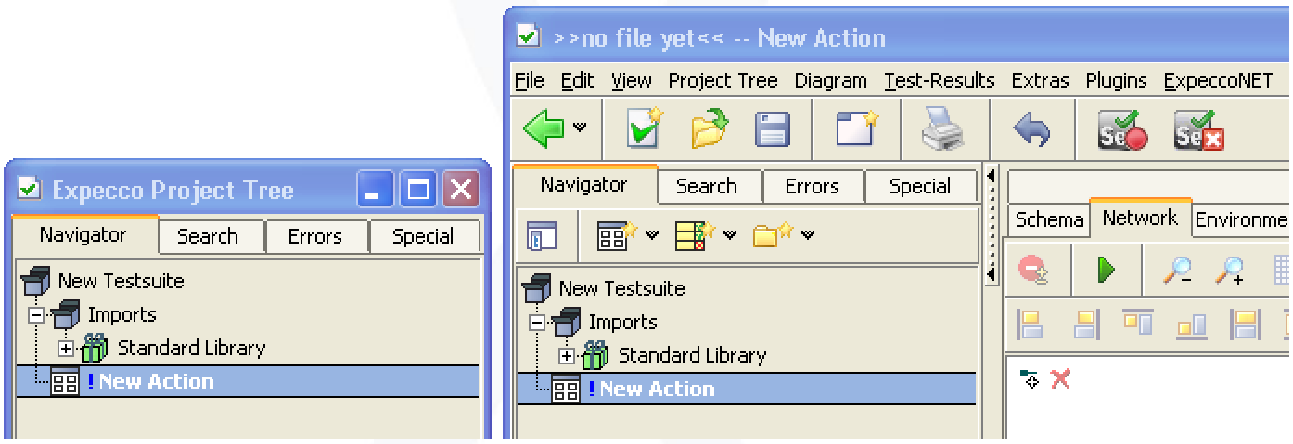
An elementary block defines a basic action, for example built-in functions, DLL-calls and JavaScript functions.

The behaviour of a compound block is represented by an activity diagram. The elements of an activity diagram are steps which are connected through data and/or control flows.

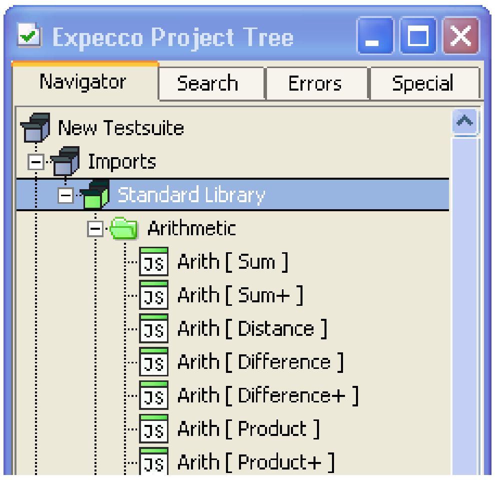
To create a new compound block click the button on the toolbar 

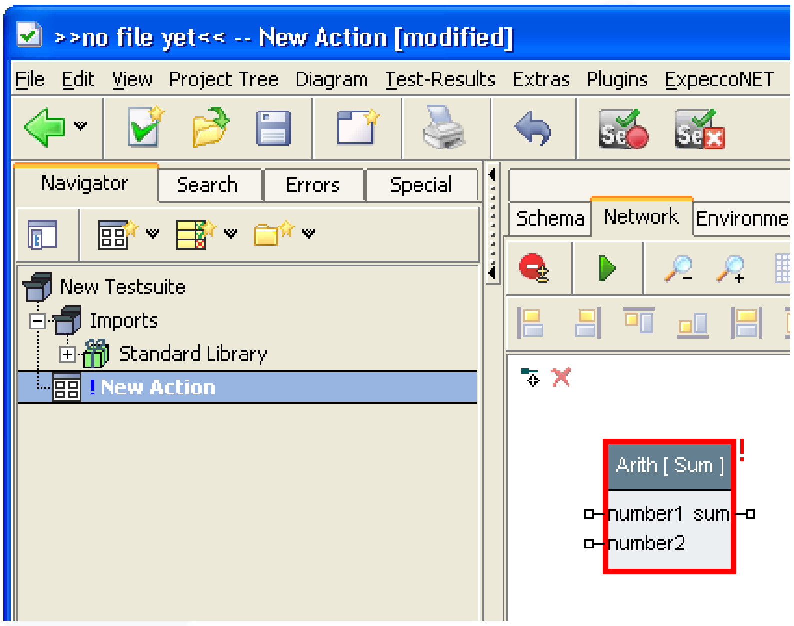

>>no file yet<< -- New Action
File Edit View Project Tree Diagram Test-Results Extras Plugins ExpeccoNET Help
Navigator Search Errors Special
New Testsuite
Imports
Standard Library
! New Action
Run
Schema Network Environment Resources History Documentation Test/Demo

Click on the button to open an additional project tree which allows you to drag elements into the network editor.

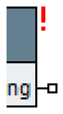

Expecco Project Tree
Navigator Search Errors Special
New Testsuite
Imports
Standard Library
! New Action
>>no file yet<< -- New Action
File Edit View Project Tree Diagram Test-Results Extras Plugins ExpeccoNET
Navigator Search Errors Special
New Testsuite
Imports
Standard Library
! New Action
Schema Network Environme

In the external tree change to the category "Arithmetic" and select the block "Arith [Sum]". Keep the mouse pressed and drag the selected element into the network of the new block. Release the mouse button and a new step will be created.

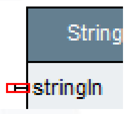




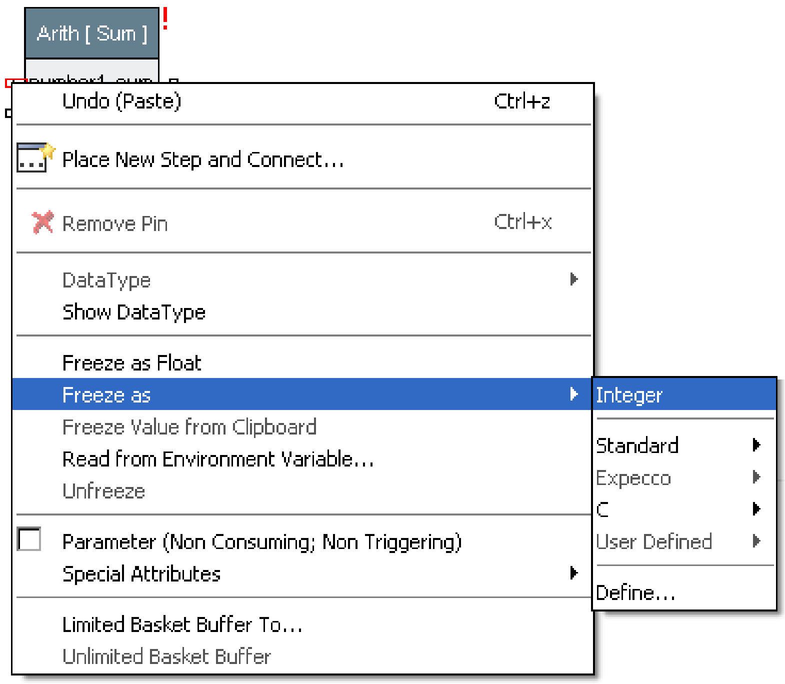
Since this step requires input values, it can not be executed (showing a red exclamation mark).



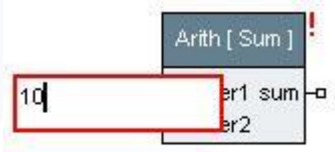
Inputs and outputs are termed as pins (connectors). In the first step we allocate the inputs with fixed values. You can do this by double clicking on the input pin



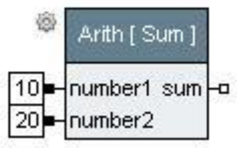
or via the corresponding entry in the context menu:



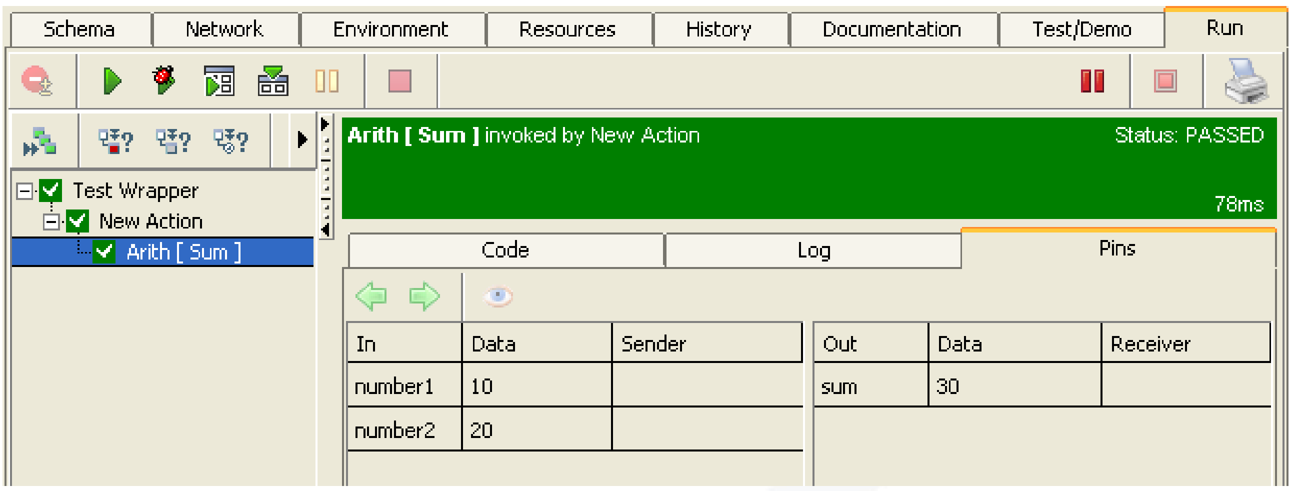
Enter any number in the box:



Allocate the second input pin. The step now has all the required input values to be executed.



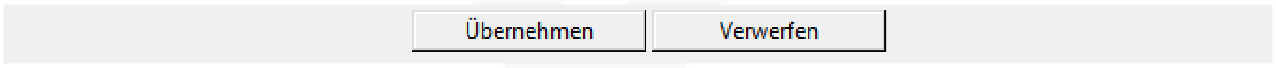
Execute your newly created block in the tab Test.



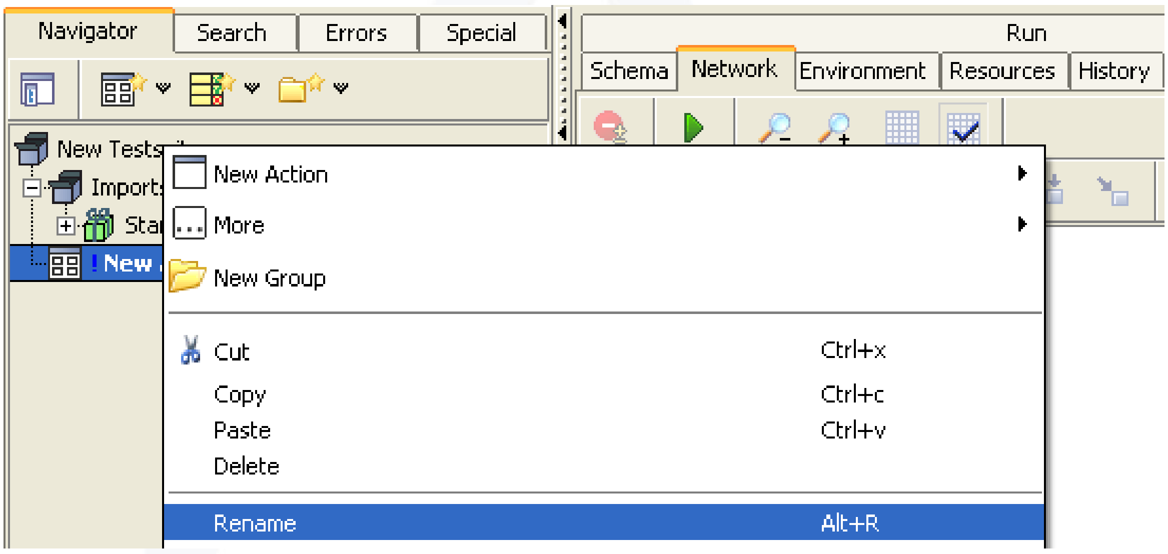
Arith [Sum] invoked by New Action — Status: PASSED — 78ms

In	Data	Sender	Out	Data	Receiver
number1	10		sum	30	
number2	20				

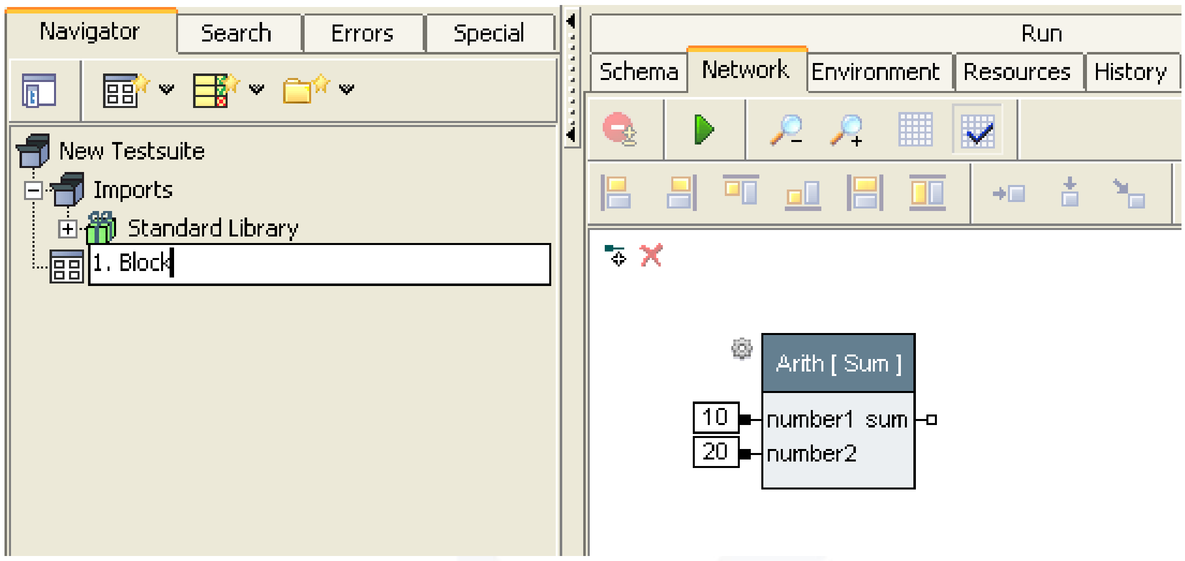
In the tab "Pins" you see the input values and the expected output value.
Accept the changes to the block in the system, click on the button "Accept".



Enter a name for the block, which can be accessed via the context menu of the newly created module in the project tree.

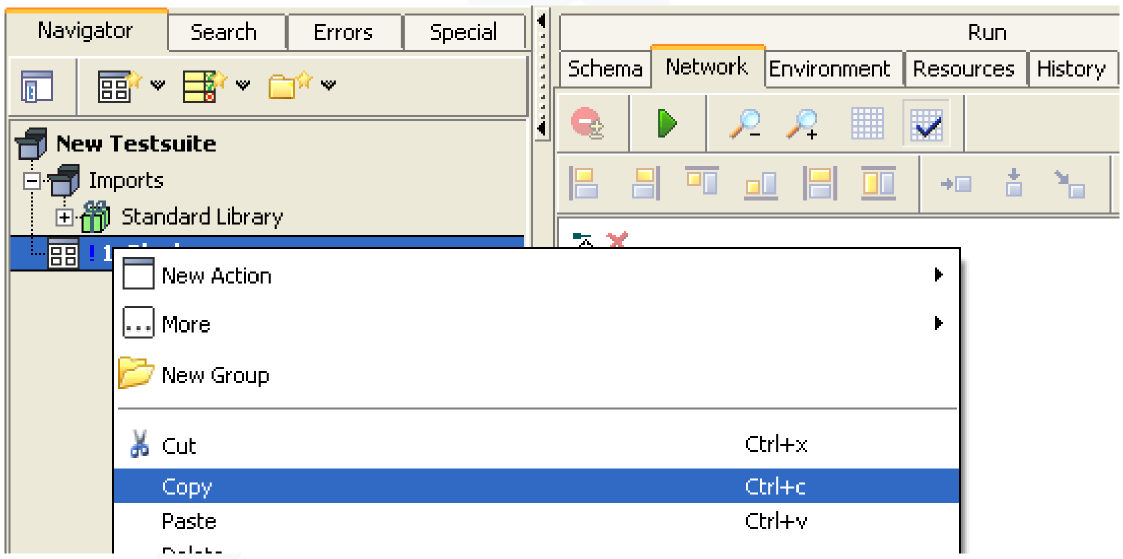


Enter the name "1.Block" in the box.

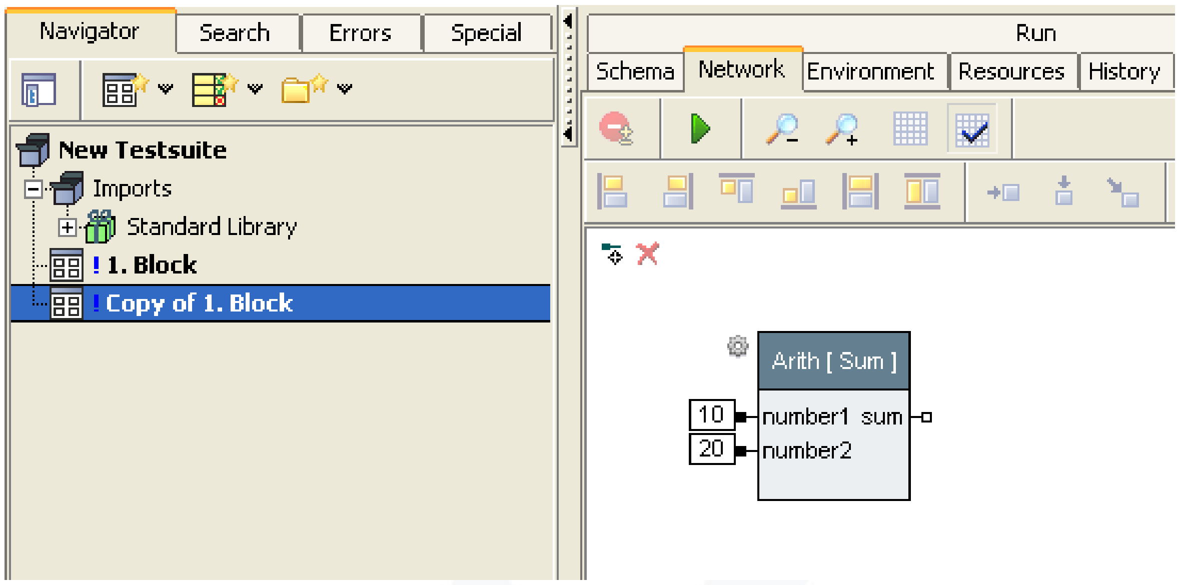


Creating a more complex block

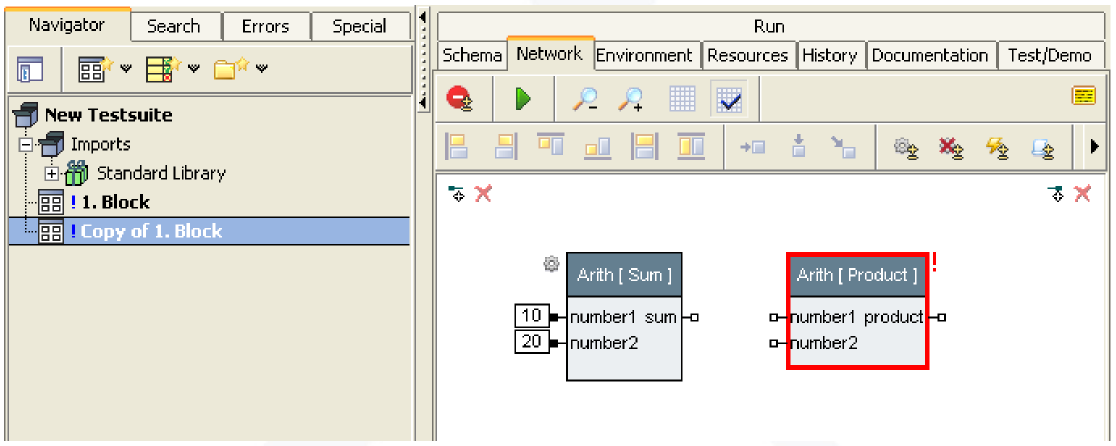
As starting point, we use a copy of the previous block. To do so, select the block and open the right-click context menu. Then select the Copy item and afterwards the Paste item.



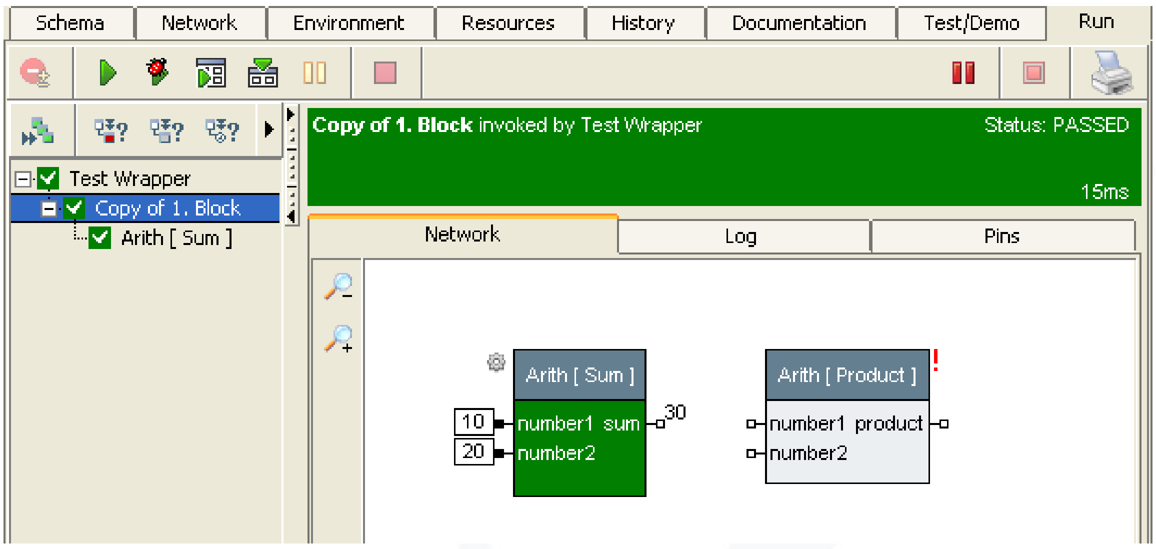
After pasting, you will have a copy of the previous block.



Place the block "Arith [Product]" from the additional tree via drag & drop into the network. Afterwards, your network should look like the following one:



After the execution of the module you will get the following picture:



Copy of 1. Block invoked by Test Wrapper   Status: PASSED

15ms

The activities will be displayed in different colors, depending on their execution state. As seen here in green - for successful. As you can also see, no activity has started for the new step. The step is lacking the required input values (recognizable through the red exclamation mark).

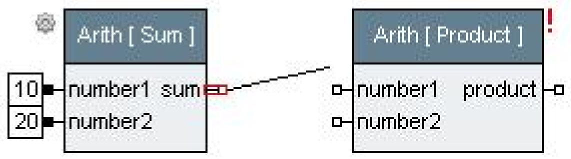
Therefore, we connect the inputs and outputs of the steps.

Most connections are data-driven, which means they have to have the same data at the beginning and at the end of the connection. For example, it is not possible to add text into an addition block.

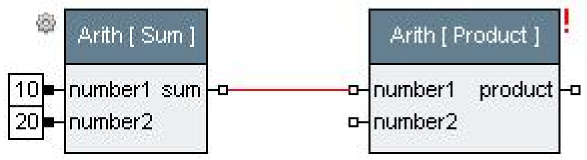
If the input and output data types do not match, various conversion blocks are available in the standard library under the category "Type Conversion".

To create a connection, follow these steps:

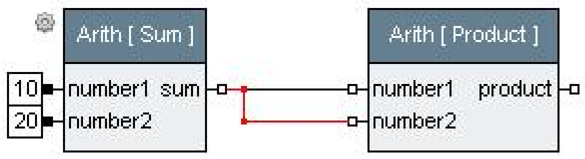
1. Select with the left mouse button, the center|output pin "sum" at the step "Arith [Sum]" and keep it pressed.



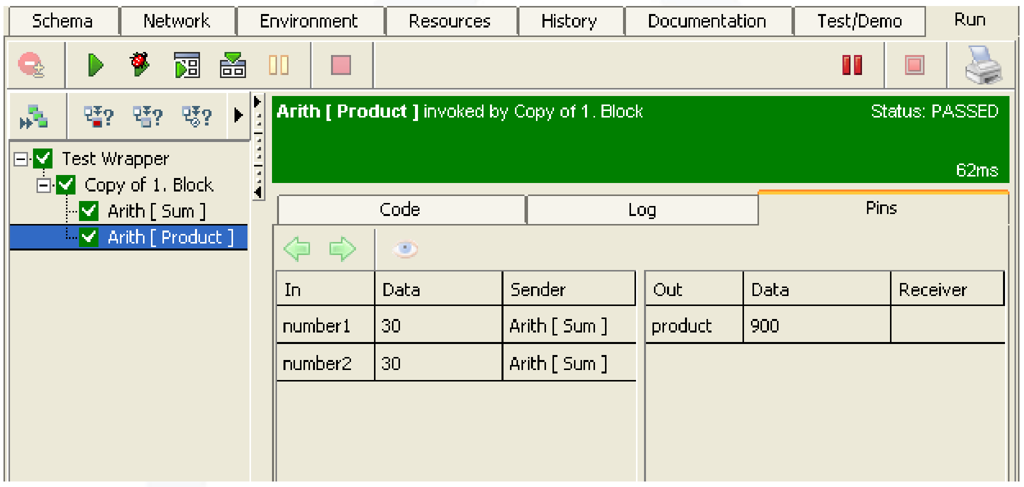
2. 2. Drag the mouse button to connect to the input pin "number1" at the step "Arith [Product]" and release the mouse button while on the input pin ⊸.



Repeat this step and connect the additional output "sum" with the input pin "number2".



Run the module again, as you can see all the steps were executed now.



Test Wrapper
Copy of 1. Block
Arith [Sum]
Arith [Product]

Arith [Product] invoked by Copy of 1. Block Status: PASSED
62ms

Code | Log | Pins

In	Data	Sender
number1	30	Arith [Sum]
number2	30	Arith [Sum]

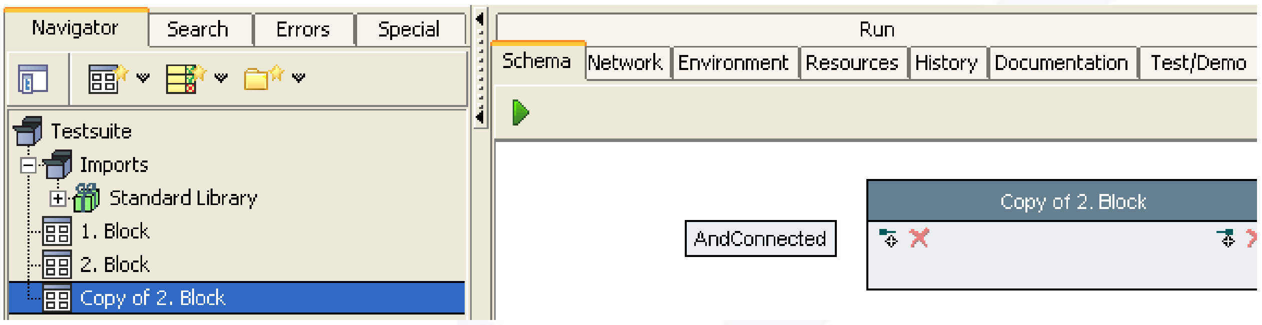
Out	Data	Receiver
product	900	

Accept the changes and rename it into "2.Block".

Adding output pins to a block

As starting point, we use a copy of the previous block, which you can create through the context menu (Copy then Paste).

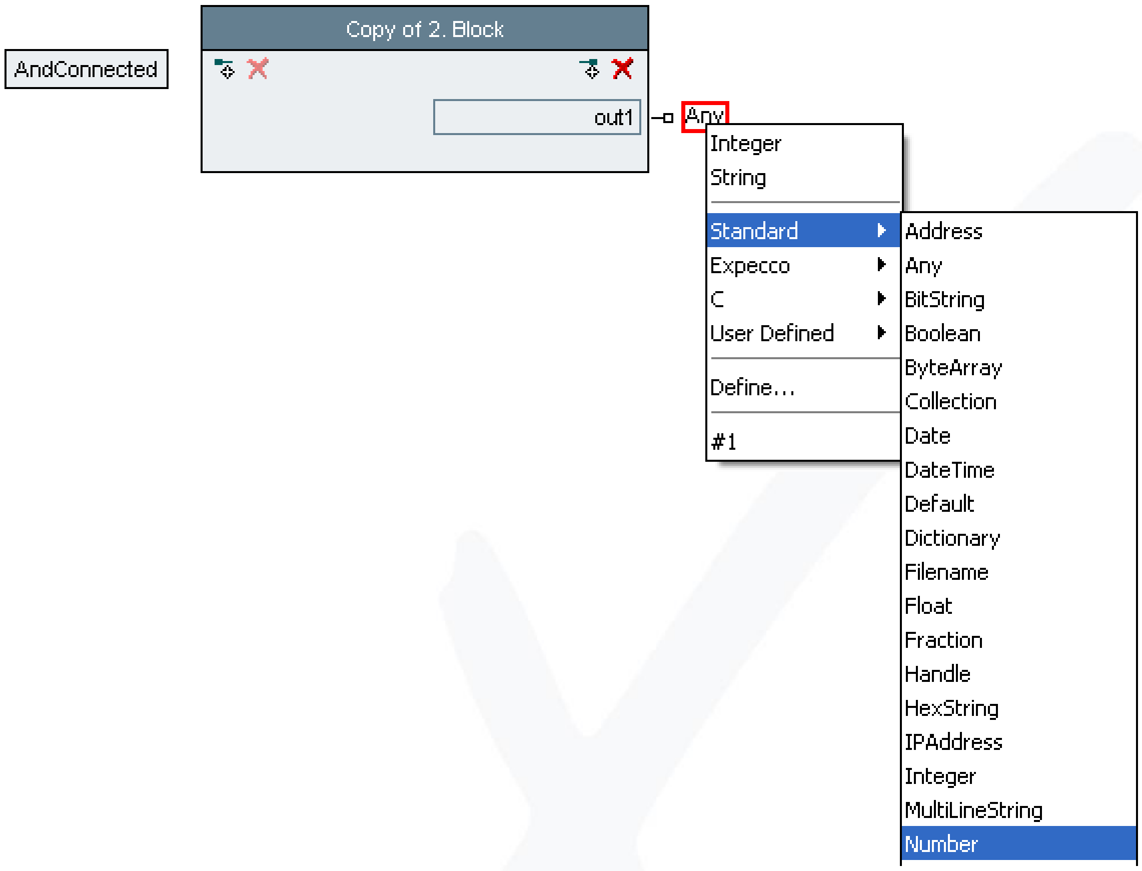
You will get the following picture:



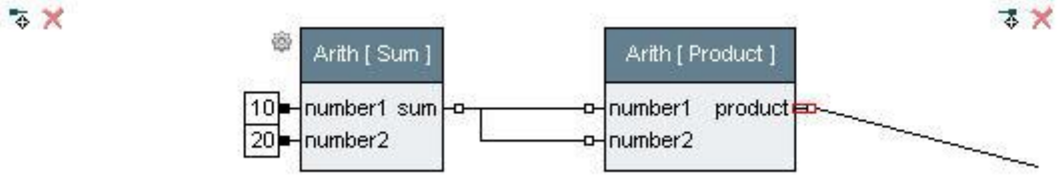
The interface of a block is described by its inputs and outputs. In the schema editor, the input pins are displayed on the left and the output pins on the right. Each pin is assigned a data type.

In order to create new pins, there are different possibilities:

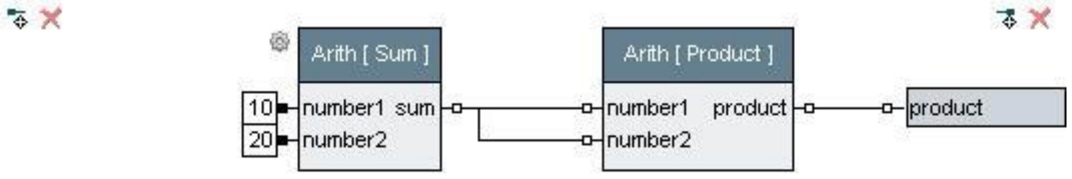
1. Click the button inside the schema editor to create a new input or output pin. To change the default data type, click on the data type and select the desired data type from the appearing menu.



2. Drag the corresponding input or output pin within the network to the left or right side of the editor.

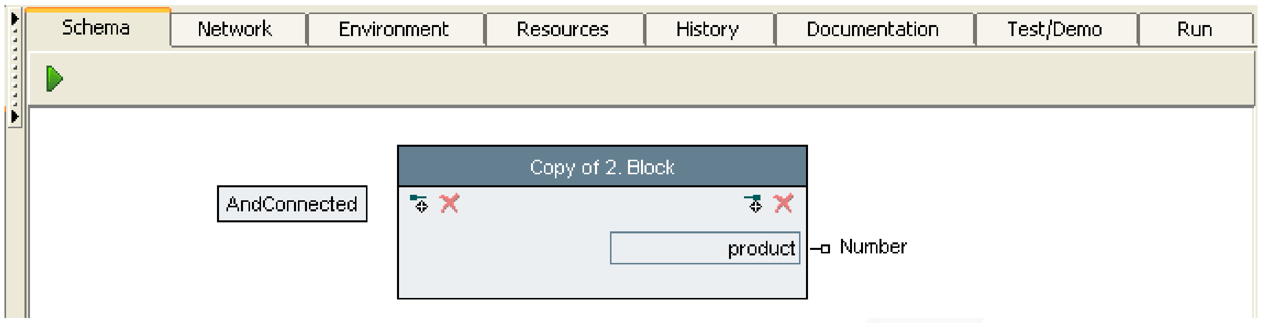


After releasing the mouse, you see the following picture at the right border:



Create an output pin as described in example 2.

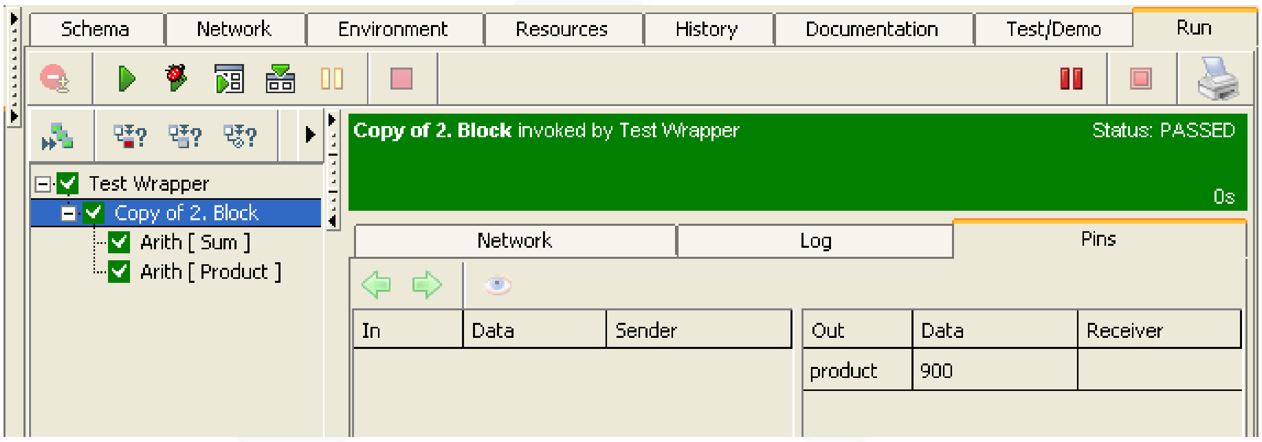
Select the tab Schema and you will get the following picture:



The block has received a new output pin. The data type and name of the output pin were taken over from the internally associated output pin within the network. By clicking on the name, the pin can be renamed.

To delete an existing input or output pin, select the appropriate pin and click on the button .

Execute the block again and you will get the following picture:



Copy of 2. Block invoked by Test Wrapper Status: PASSED 0s

In	Data	Sender

Out	Data	Receiver
product	900	

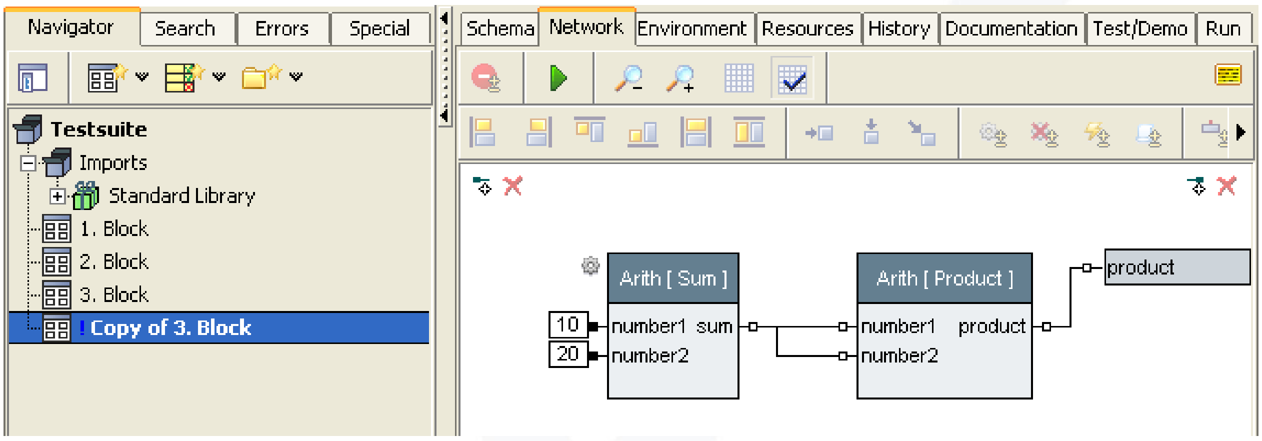
The output value is now available for further use at the output pin.

Accept the changes and rename it into "3. Block".

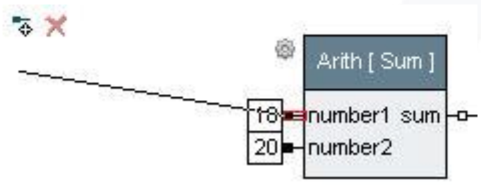
Adding input pins to a block

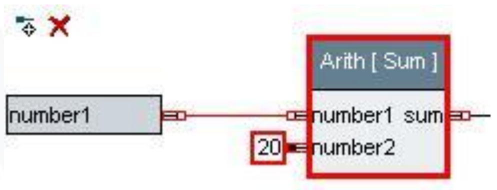
As starting point, we use a copy of the previous block, which you can create through the context menu (Copy then Paste).

You will get the following picture:

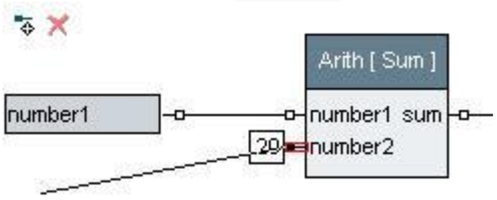


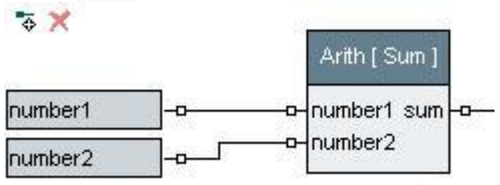
To externally supply the step "Arith [Sum]", create the input pins (same approach as for the Output pins):



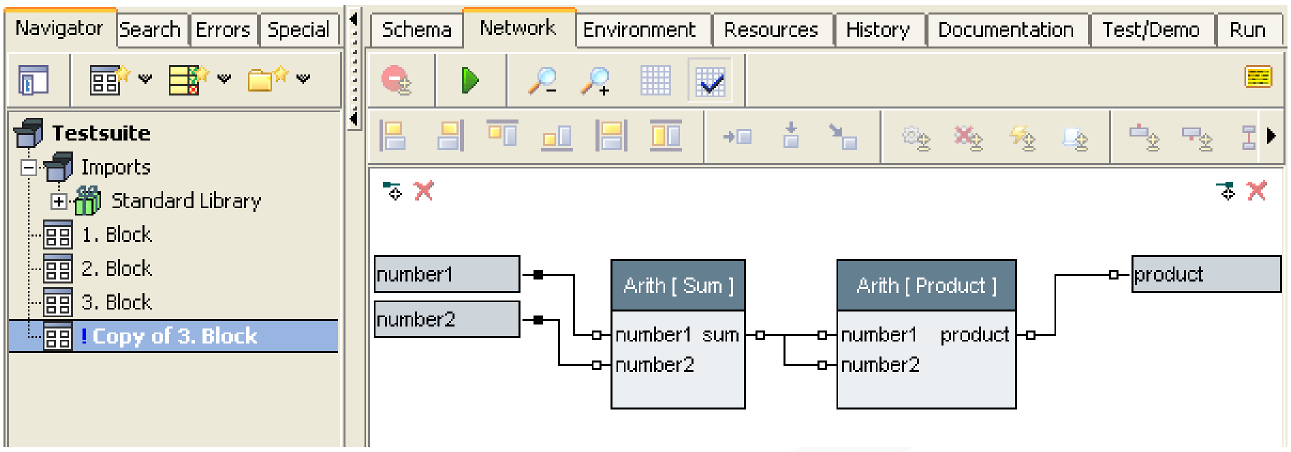


Perform the same action for the second input pin "number2":

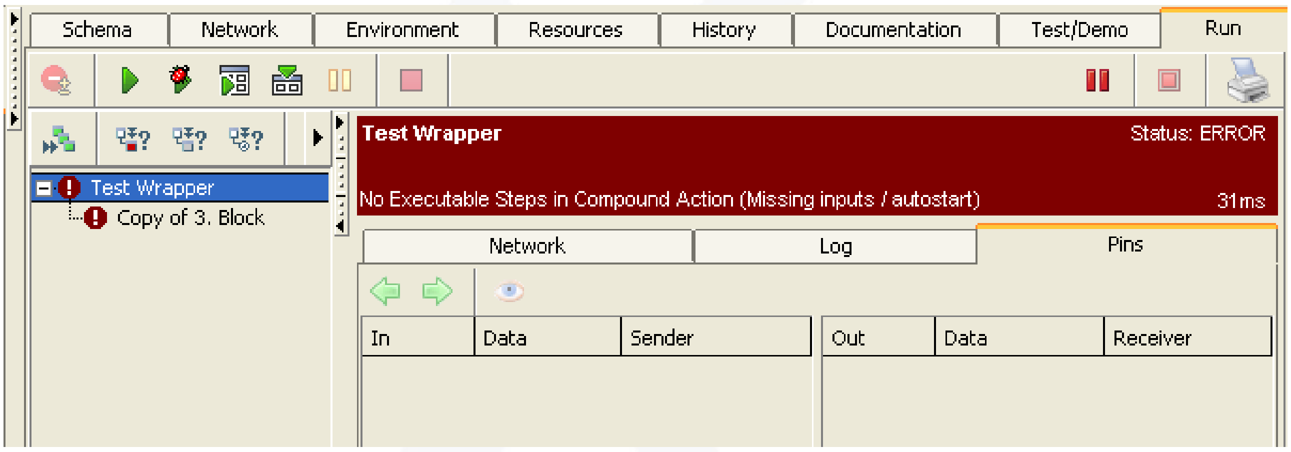




You will get the following picture:



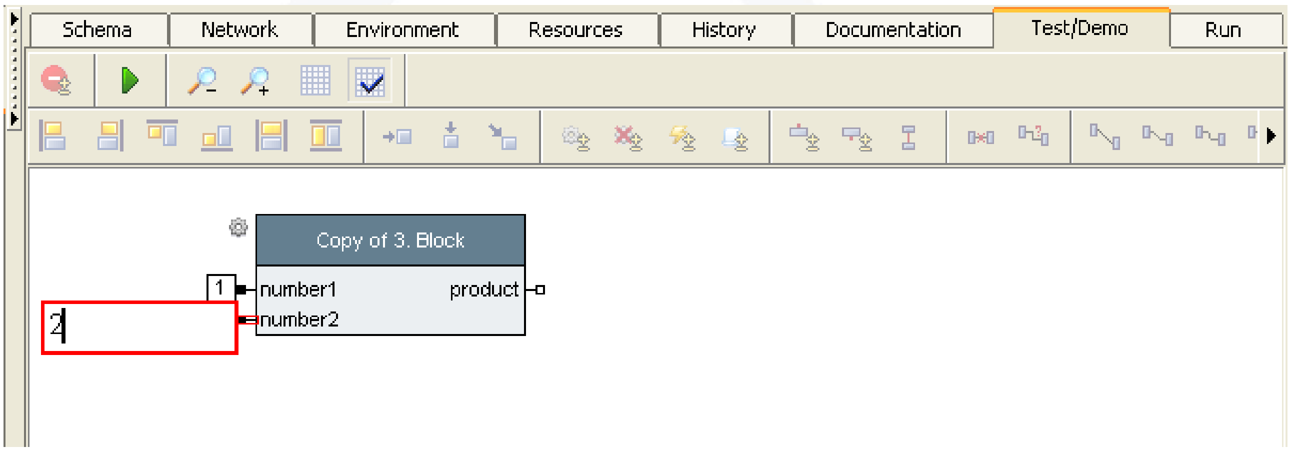
Execute the block again and you will get the following picture:



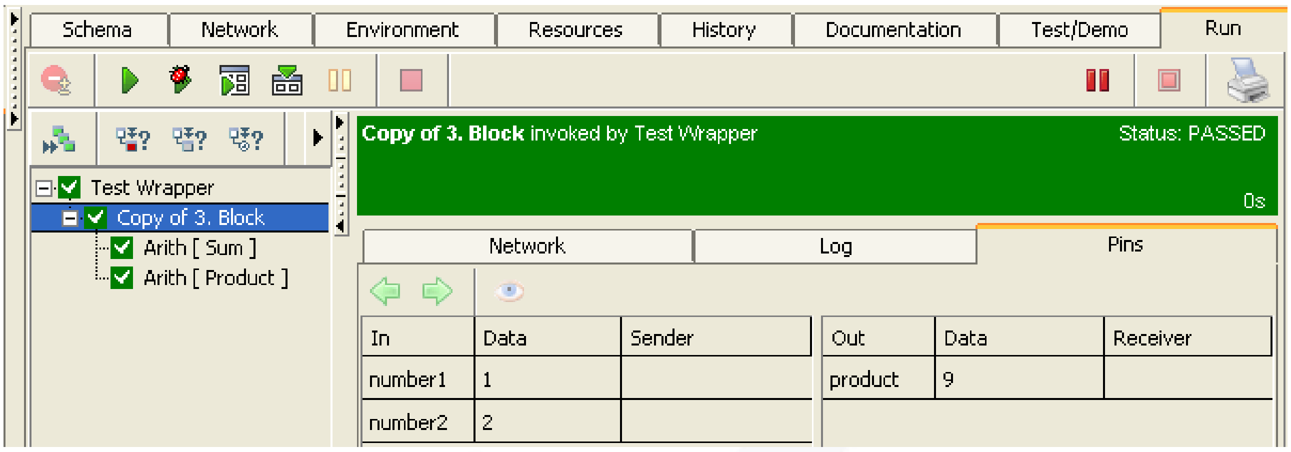
Test Wrapper Status: ERROR

No Executable Steps in Compound Action (Missing inputs / autostart) 31ms

As you can see, the block could not be executed because the necessary input values for the input pins of our test module are missing. Freeze the input with fixed values:



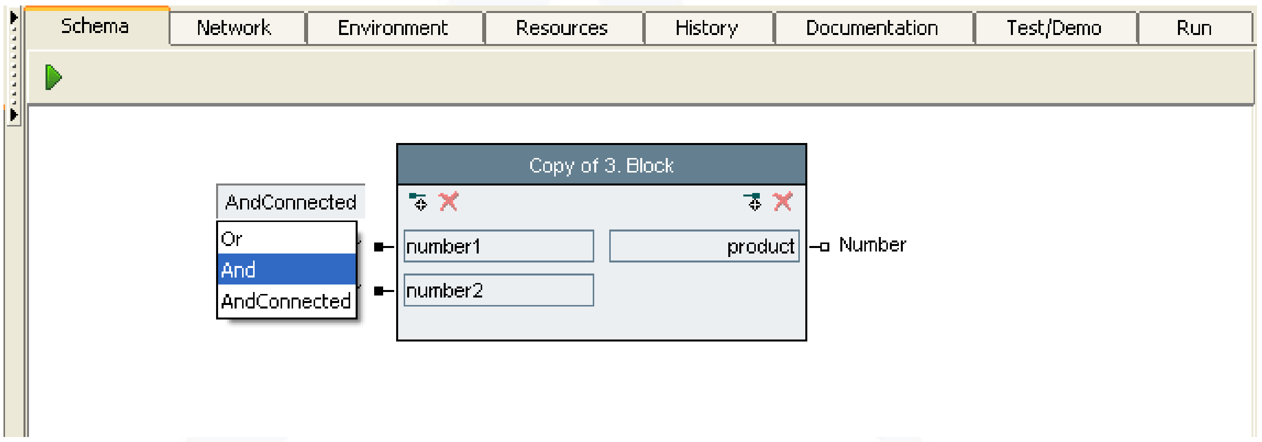
Execute the block again and you will get the following picture:



Copy of 3. Block invoked by Test Wrapper   Status: PASSED   0s

In	Data	Sender	Out	Data	Receiver
number1	1		product	9	
number2	2				

Now go to the tab Schema. Here you can change the trigger conditions for the block. Click on the field "And-Connected". A submenu is displayed on the configuration:



Here you can specify which input pins must have values in order for the block to be executed. Three options are supported:

1. And-Connected

All connected input pins must have values, open inputs are ignored.

2. And

All input pins must have values.

3. Or

At least one input pin must have a value.

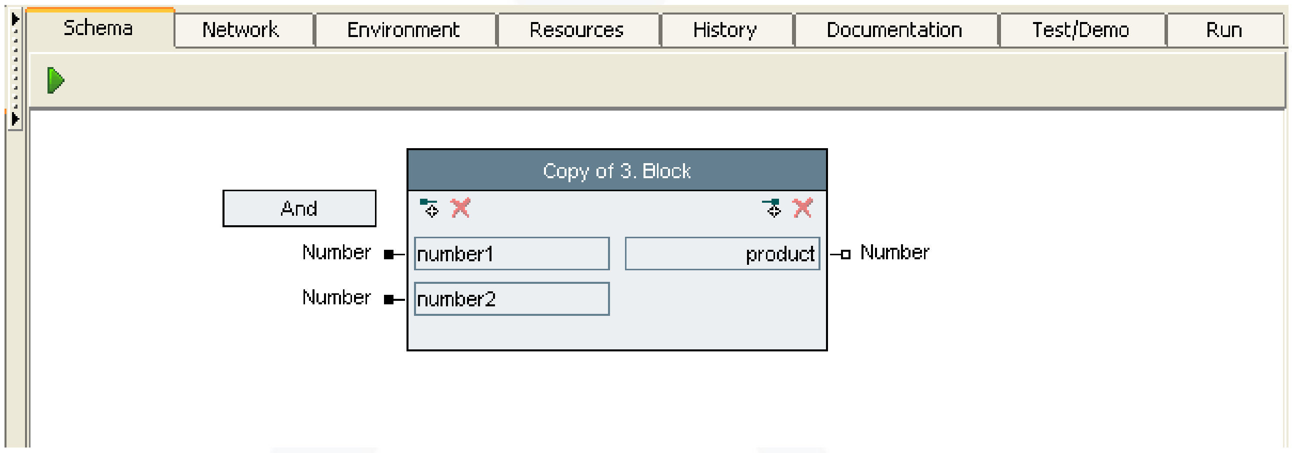
In our case, all input values have to be supplied, so choose the "And" condition.

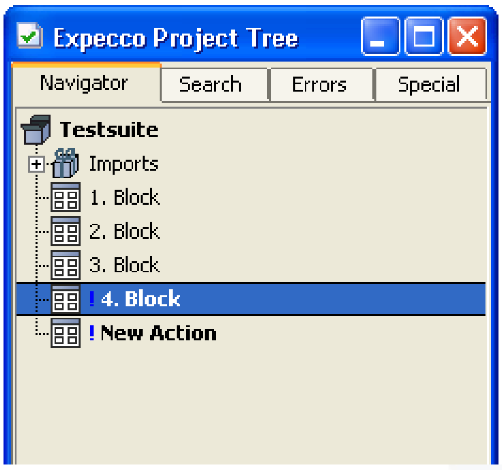
Accept the changes and rename it into "4. Block".

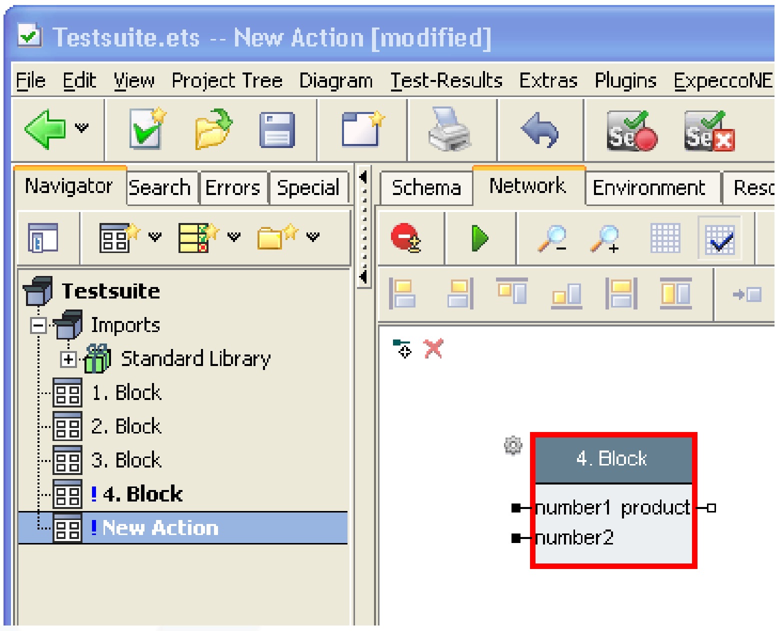
Reuse of blocks

The way of working with blocks is the key to reusability and flexibility. Once a block is created it can be reused in any compound block.

Insert a new compound block by clicking the button . Now drag & drop the previously created blocks into the network (activity diagram) of the new block.



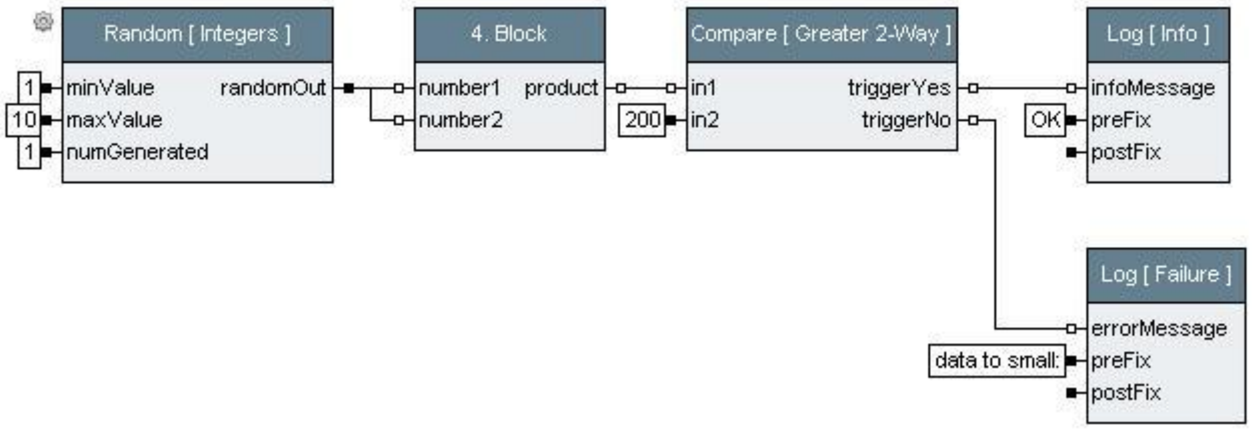




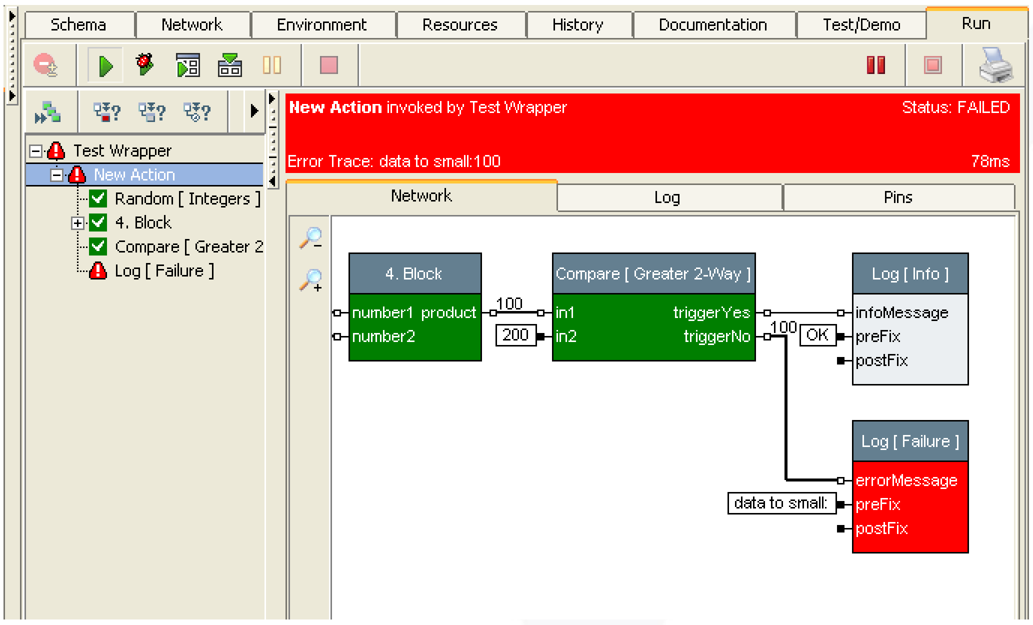
Add additional blocks from the Standard Library and create connections as shown in the following activity diagram.

You can find the blocks under the following categories:

- Data Generators
  Random [Integers]
- Compare
  Compare [Greater 2-Way]
- Assertions, Exceptions & Logging
  Log [Info], Log [Failure]



Due to the upstreamed "Random [Integers]" step, the created "4. Block" is supplied by different values, leading to different results. This can be seen best, when you execute the block in the Test / Demo tab a few times.



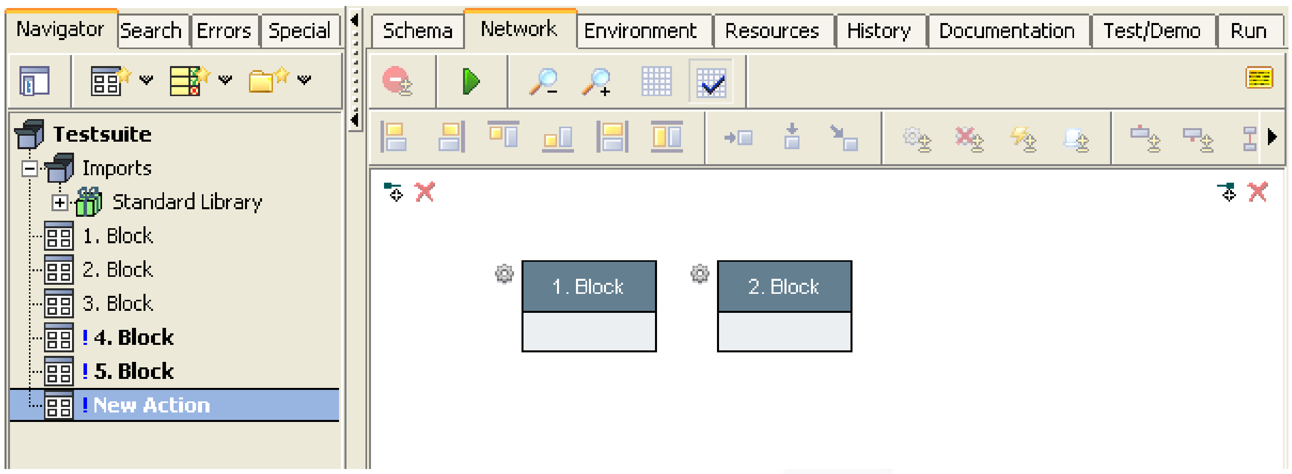
New Action invoked by Test Wrapper   Status: FAILED
Error Trace: data to small:100   78ms

Accept the changes and rename it into *"5. Block"*.

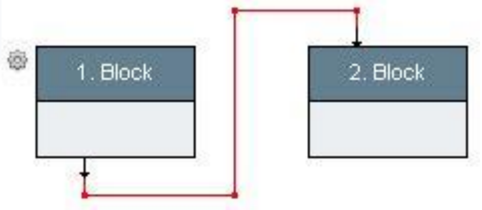
Sequential execution of steps

For a graph with no data flows (i.e., with exclusive control flows), trigger inputs will be connected with trigger outputs, creating a pure flow chart.

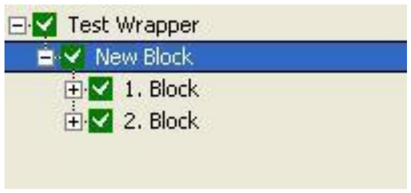
To prepare, create a new compound block by clicking the button . Drag and drop the blocks 1 and 2 in the network and you will get the following picture:



Select the "1. Block". By holding the shift key you can add the "2. Block". Both steps are now selected in the order, in which they should be executed sequentially. By clicking the button , a control flow connection from step 1 to step 2 is set.

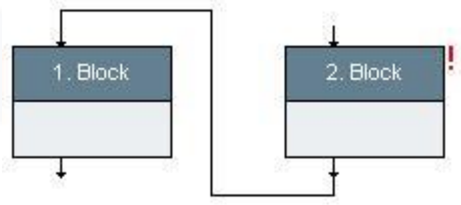


After executing the block in theTest tab, you will get the following picture. As you can see "1. Block" was executed before "2. Block".

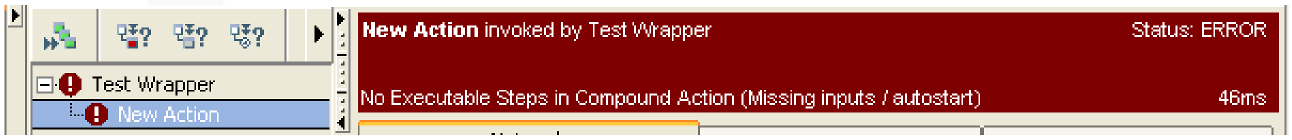


You can delete a connection by clicking it and afterwards select remove or press the button .

Now, delete the existing control flow connection and set up the control flow in the reverse direction. Proceed as described above, only selecting "2. Block" before "1. Block" this time.



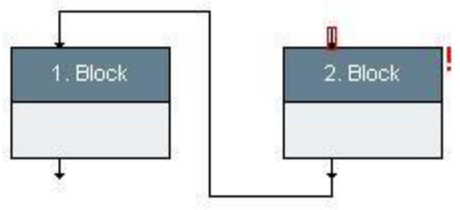
After executing the block, you will now get the following error screen:



New Action invoked by Test Wrapper Status: ERROR

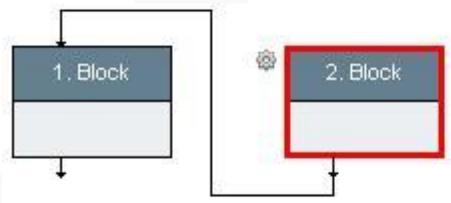
No Executable Steps in Compound Action (Missing inputs / autostart) 46ms

As you can see the in the network editor, the "2. Block" is marked with a red exclamation mark, which means it is not ready to be executed.

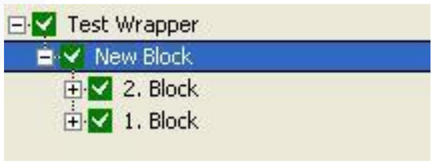


The step has an open trigger input pin. By connecting the trigger input with any other output pin, the action of the step can also be triggered.
In our case, delete the trigger input by selecting it and then press the Delete key.

To execute a step, it has to have at least one connected not-parameter pin or the auto-start attribute. Therefore, mark step 2 as an auto-start. To do so, select it and click on the button



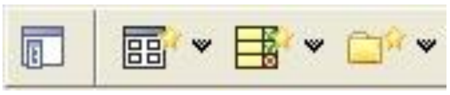
After executing the block in theTest tab, you will see the following picture. As you can recognize the "2. Block" was executed before the "1. Block".



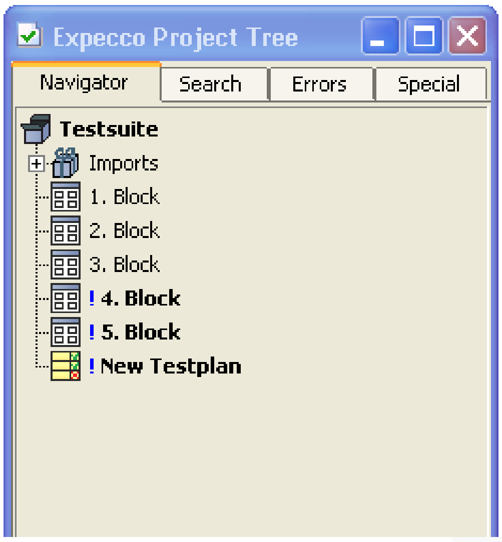
Creating a testplan

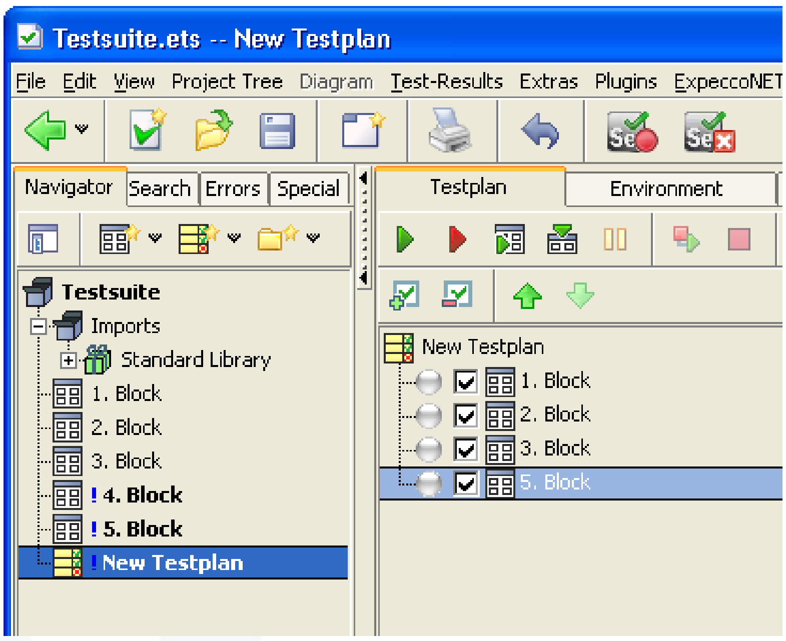
In expecco, a collection of several testcases is described as a testplan. During execution of a testplan, all individual testcases are executed sequentially.

To generate a new testplan, click the button on the toolbar.



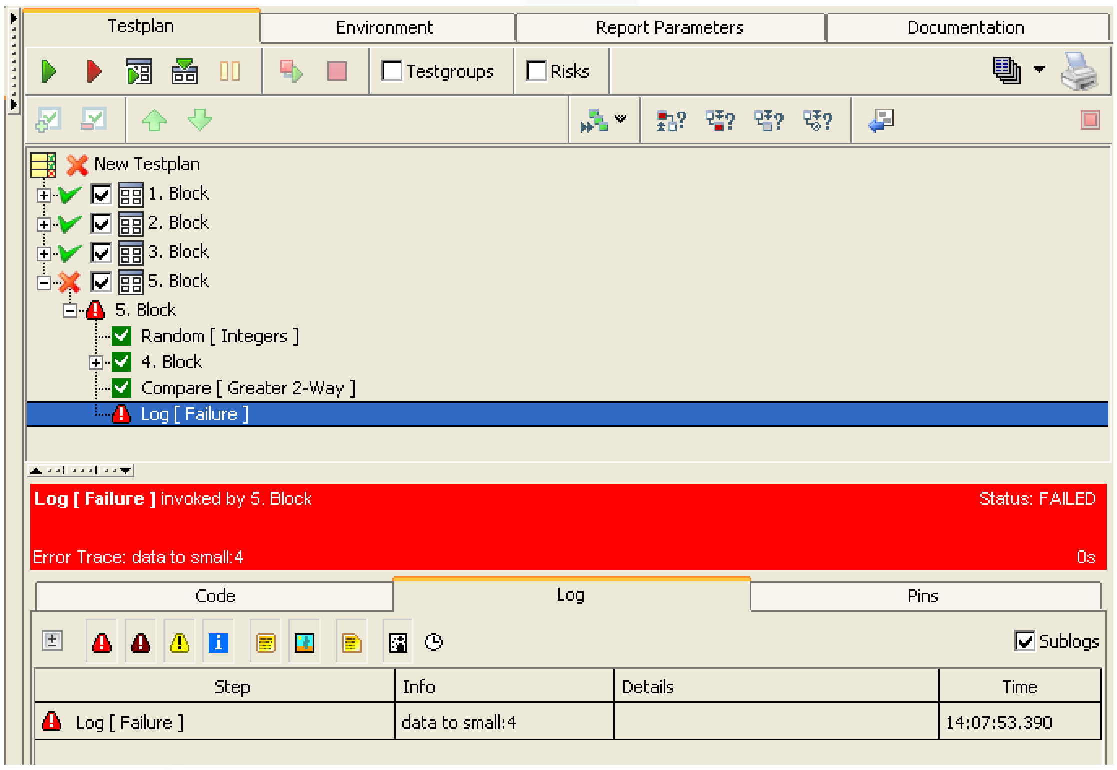
Drag & drop the newly created blocks, except for "4. Block", into the test plan editor. You will get the following picture:





Accept the changes and execute the testplan by pressing the button ▶

You will get the following picture:



```
New Testplan
  ✓ 1. Block
  ✓ 2. Block
  ✓ 3. Block
  ✗ 5. Block
      5. Block
        ✓ Random [ Integers ]
        ✓ 4. Block
        ✓ Compare [ Greater 2-Way ]
          Log [ Failure ]
```

Log [Failure] invoked by 5. Block   Status: FAILED

Error Trace: data to small:4   0s

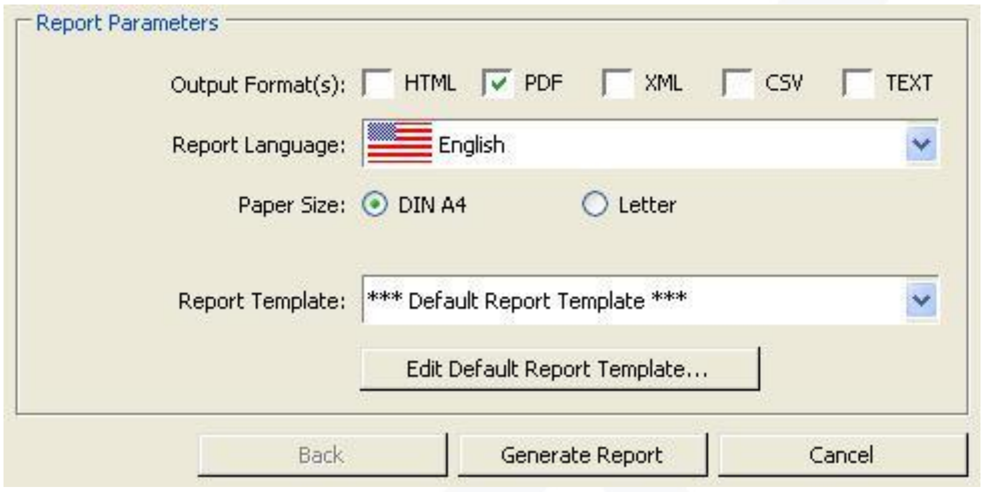
Step	Info	Details	Time
Log [Failure]	data to small:4		14:07:53.390

Since the "5. Block" works internally with randomly generated input values, you will get different results with multiple executions for the testplan.

Generating a report

For preparation, execute the testplan. To create and view a detailed report about the test execution, click on the button

The dialog box "Report Generation / Print" will open. Prior to the report generation, you will have to determine further settings:



Report Parameters

Output Format(s): HTML [✓ PDF] XML CSV TEXT

Report Language: English

Paper Size: [✓ DIN A4] Letter

Report Template: *** Default Report Template ***

- Output Format(s):
  The reports can be created in different output formats. A multiple selection is possible.

- Report Language:
  Set the language for the report.

- Paper Size:
  Only for the PDF output format the paper size is specified. For all other formats this setting

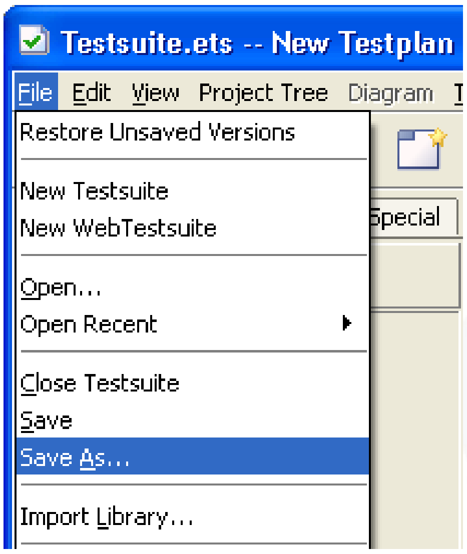
  will be ignored.

- Report Template:
  Here, the content and appearance of the report can be determined.
  For more information, go to Report Generation.

Click on the button "Generate Report" to start the report generation. After generating the file is open within a browser.

Saving the testsuite

To save the testsuite, click into the menu "File" → "Save As...". The Save dialog will appear.



Summary

In our example, we have intentionally used simple blocks from the Standard Library to help you work with expecco.

However, this example is unrealistic in that no System Under Test (SUT) has been supplied with signals or been tested. As the interface to a SUT is often device and protocol specific, the interfaces are not part of the expecco base system. They are realized by specific libraries and plugins. These can be supplied by eXept or by third parties. You can also create and maintain them yourself.

Information on specific protocol blocks (TCP / IP, DLL calls, FTP, ...) and interfaces to specific devices can be found in the documentation of the various libraries and plugins.

You can find this information in the menu:

- Help
- expecco Wiki