Get into GNED

An introduction to the GNED editor of OMNEST^{TM}/OMNET++.
By Gábor Tabi

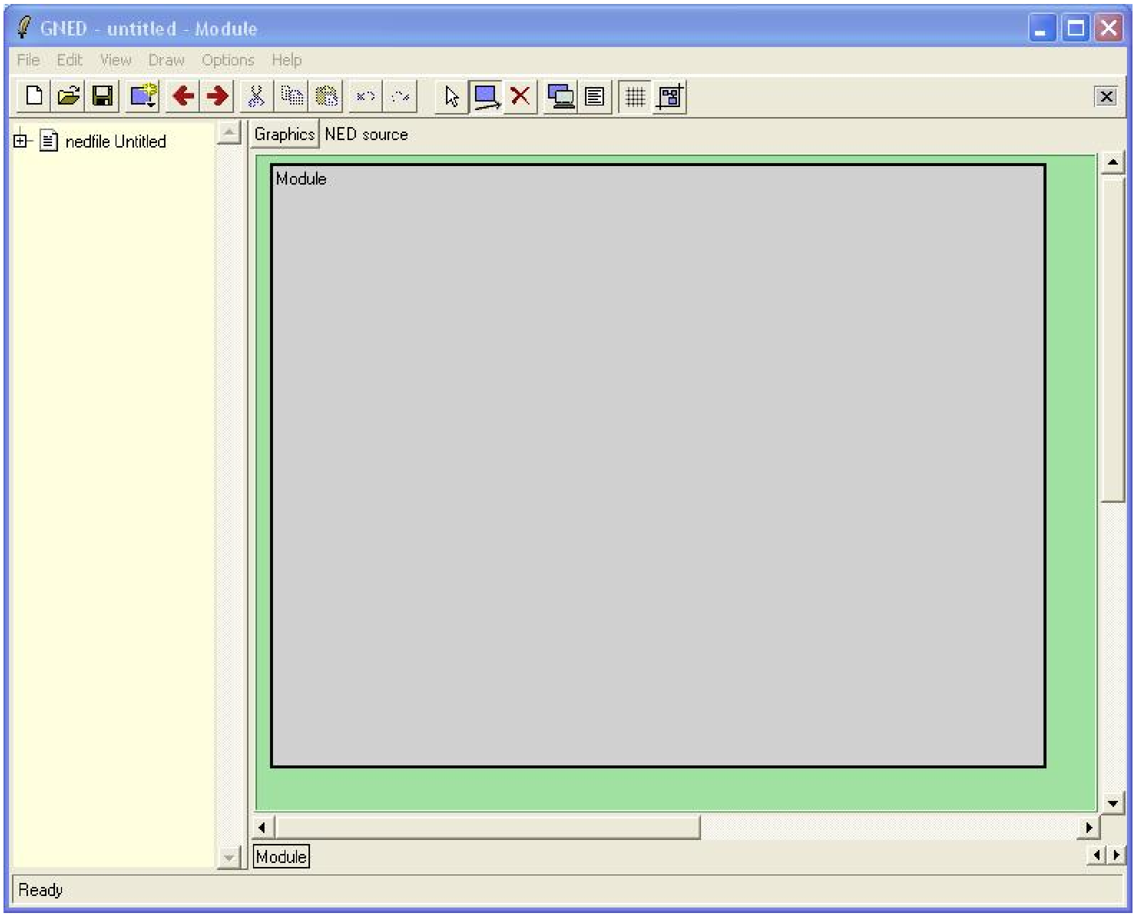
Table of Contents

How to use this manual

This manual is intended to provide an introduction to the use and features of GNED. The development environment that is part of the OMNEST^{TM}/OMNET++ simulation package. GNED is a graphical environment, facilitating the development and debugging of simulation topologies. The graphical and textual representations of model topologies can be used interchangeably, so editing, visualizing and parameter entry may take place in either the text or graphical views.

1. Introducing the interface

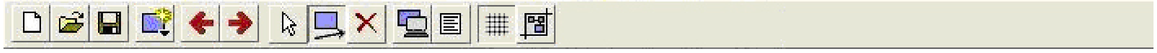
Let us first examine the interface, as it first appears on your screen.



GNED supports two „views” of your model topology: graphical editing view, which is the default view, when the software starts up, and „NED source view”, where you can edit the source code directly. Switching between these modes is performed by pressing either of the buttons above.

The command bar is different for „Graphics” and „NED source views”, commands that can not be applied to the current view are hidden to simplify the interface.

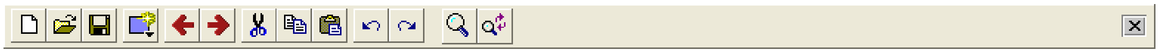
The „Graphics” command bar bellow the menu bar contains the following icons, and functions:



1. 🗋 New NED file: Creates a new NED document - you can have multiple NED files open at a time.

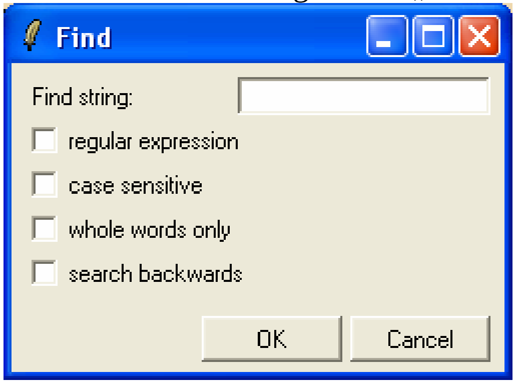
2. Open NED file: the new NED file will be editable, and visible in the document NED component tree view.
3. Save Document: Saves the currently active NED file.
4. Add new component to current NED file: A dropdown list will appear upon releasing this button. You can select from four types of components to add to your NED file: Import, Channel, Simple Module, Compound Module or Network
5. Back: Navigates inside the history of your views, backwards.
6. Forward: Navigates inside the history of your views, forwards.
7. Select, move or resize items button. You can select and manipulate items in graphics mode, if this tool is active. To resize an item, drag its corner. If the item is represented by an Icon, you can not resize it this way. To move it, click inside it and drag.
8. Draw submodules and connections tool. To draw a submodule, click and drag inside of a module. To connect two modules (one way!) click inside a submodule and drag the rubber-band into another submodule. To create two-way connections, you will have to create a one-way connection twice, in opposite directions.
9. Appearance of selected items: Clicking this button will bring up the „Submodule Appearance", „Module Appearance" or „Connection Appearance" dialogs, depending on which type of object is currently selected. You must have one (and only one) module or submodule selected for this function to work. The dialog and its functions are described in more detail later.
10. Properties of selected items
11. Snap to grid on/off switch
12. Fit compound module area to contents: Pressing this button will automatically expand the boundaries of the containing module to fit the boundaries of all the submodules on screen.

The „Graphics" command bar bellow the menu bar contains the following icons, and functions:



1. New NED file: Creates a new NED document - you can have multiple NED files open at a time.
2. Open NED file: the new NED file will be editable, and visible in the document NED component tree view.
3. Save Document: Saves the currently active NED file.

4. Add new component to current NED file: A dropdown list will appear upon releasing this button. You can select from four types of components to add to your NED file: Import, Channel, Simple Module, Compound Module or Network
5. Back to previous module: Makes the previous module active.
6. Forward to next module: Makes the next module active.
7. Cut: Cut to clipboard.
8. Copy to clipboard.
9. Paste from clipboard.
10. Undo.
11. Redo.
12. Find text.
    This function will bring out the „Find Text” dialog.

    Find

    Find string:

    □ regular expression

    □ case sensitive

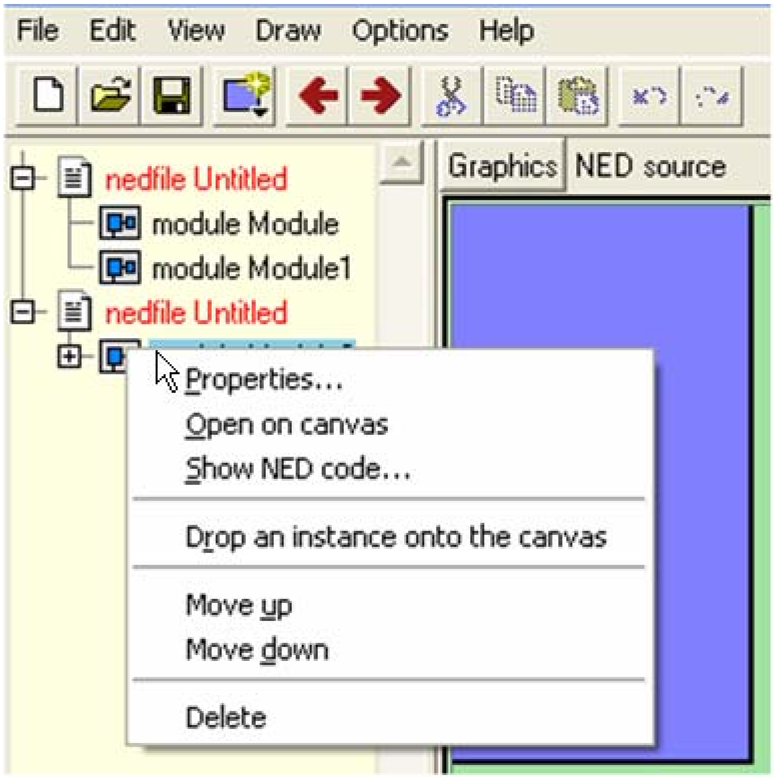
    □ whole words only

    □ search backwards

    OK Cancel

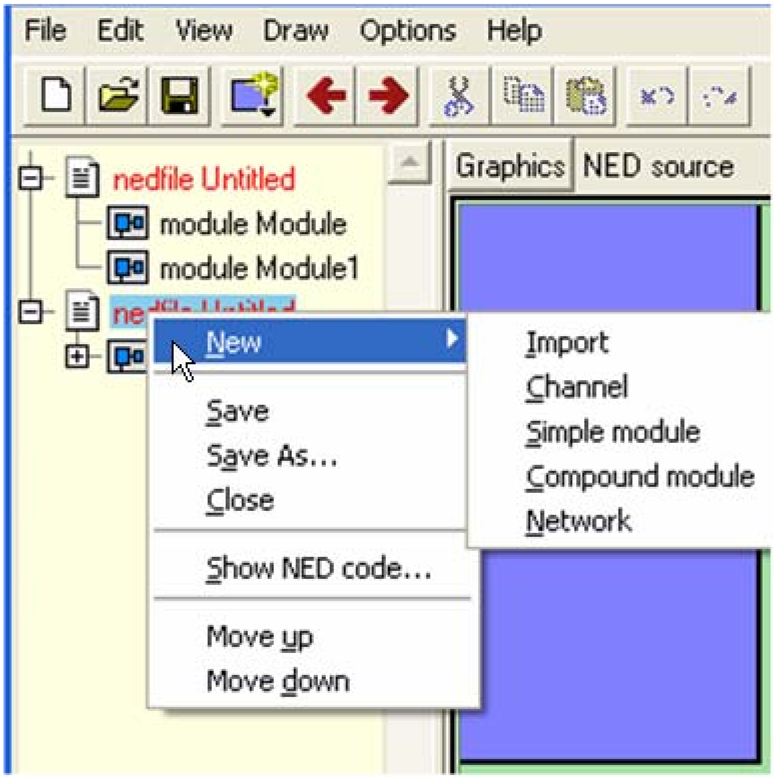
    You can specify the string you are looking for, and set filter options to search for a specific type of text only.

The left pane is a „Tree View” of all the NED files and Modules you have open in the editor at a time. You can expand any of them, and also access the context sensitive menu items relevant to each item by right-clicking on them.



The two "nedfile Untitled" items are marked red by GNED, because they are not yet saved.

If you right-click on a NED file instead, the context menu will contain the operations relevant to NED files. This way, the tree view allows you to quickly navigate and manipulate your entire simulation across NED files and modules.

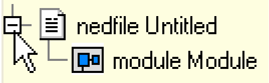


The New command will insert the element you chose from the menu into the NED file you right-clicked on.

The central pane contains either the graphical representation of your current module, or the NED source of it, depending on the option you selected on the view bar.

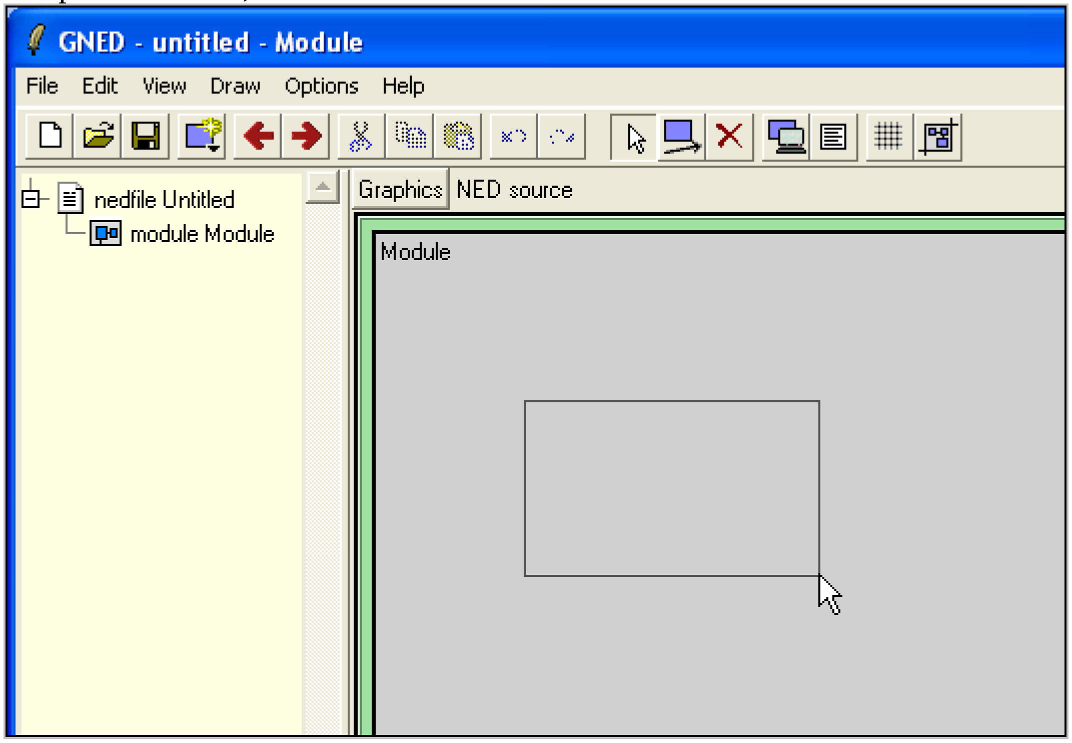
Let us quickly examine the options available in Graphics view. GNED will help you get acquainted with the interface by popping up hints initially, explaining some of the more complex functions of the interface. You can turn this off later, by checking the appropriate box. They can only be made to reappear, if you delete the .gnedrc file, so think twice before turning them off.

When you first start GNED, you will just have a single NED file open and visible in the tree view called „nedfile Untitled", and upon expanding it, you will notice, that it contains only a single Compound Module.

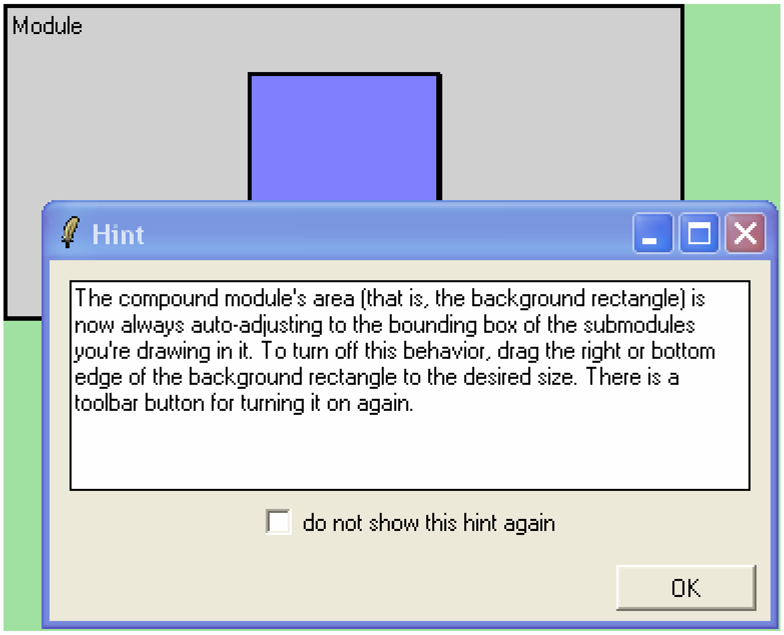


2. Using the Interface to Perform Basic Actions.

Let us click and drag inside the grey module area. This will create a submodule inside the compound module, like so:



GNED will create the submodule, and the size of the encompassing module will be automatically adjusted. You will be presented with a hint:

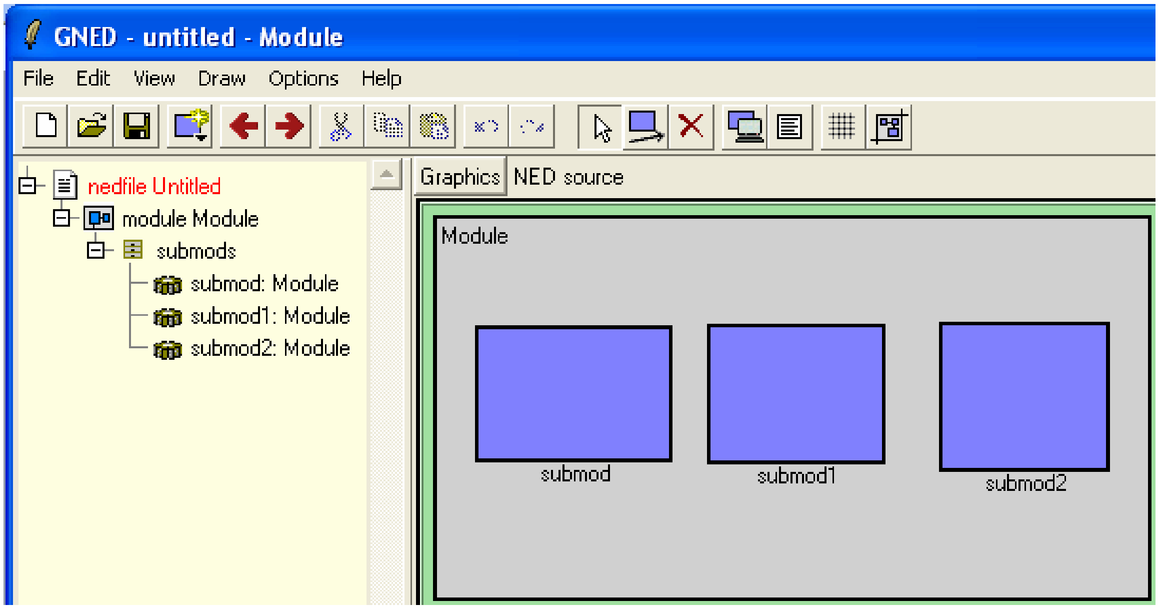


The compound module's area (that is, the background rectangle) is now always auto-adjusting to the bounding box of the submodules you're drawing in it. To turn off this behavior, drag the right or bottom edge of the background rectangle to the desired size. There is a toolbar button for turning it on again.

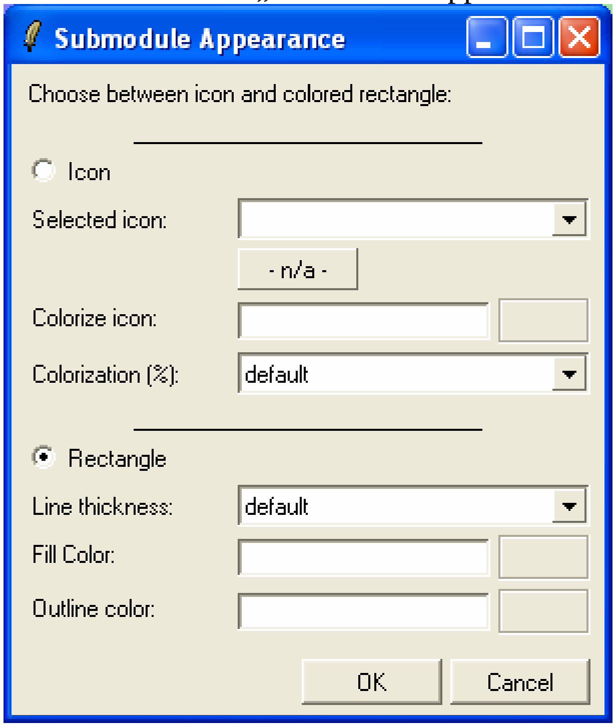
□ do not show this hint again

OK

Press OK. You have created your first submodule. Proceed to create two more, you should end up with something like this:



Now select one of them, (mode!). The selected submodule will be highlighted in red, and press the (Appearance) button, or use the context sensitive menu by right-clicking on „submod” and chosing „Appearance...”.
You should see the „Submodule Appearance” dialog.



Choose between icon and colored rectangle:

○ Icon

Selected icon:

- n/a -

Colorize icon:

Colorization (%): default

● Rectangle

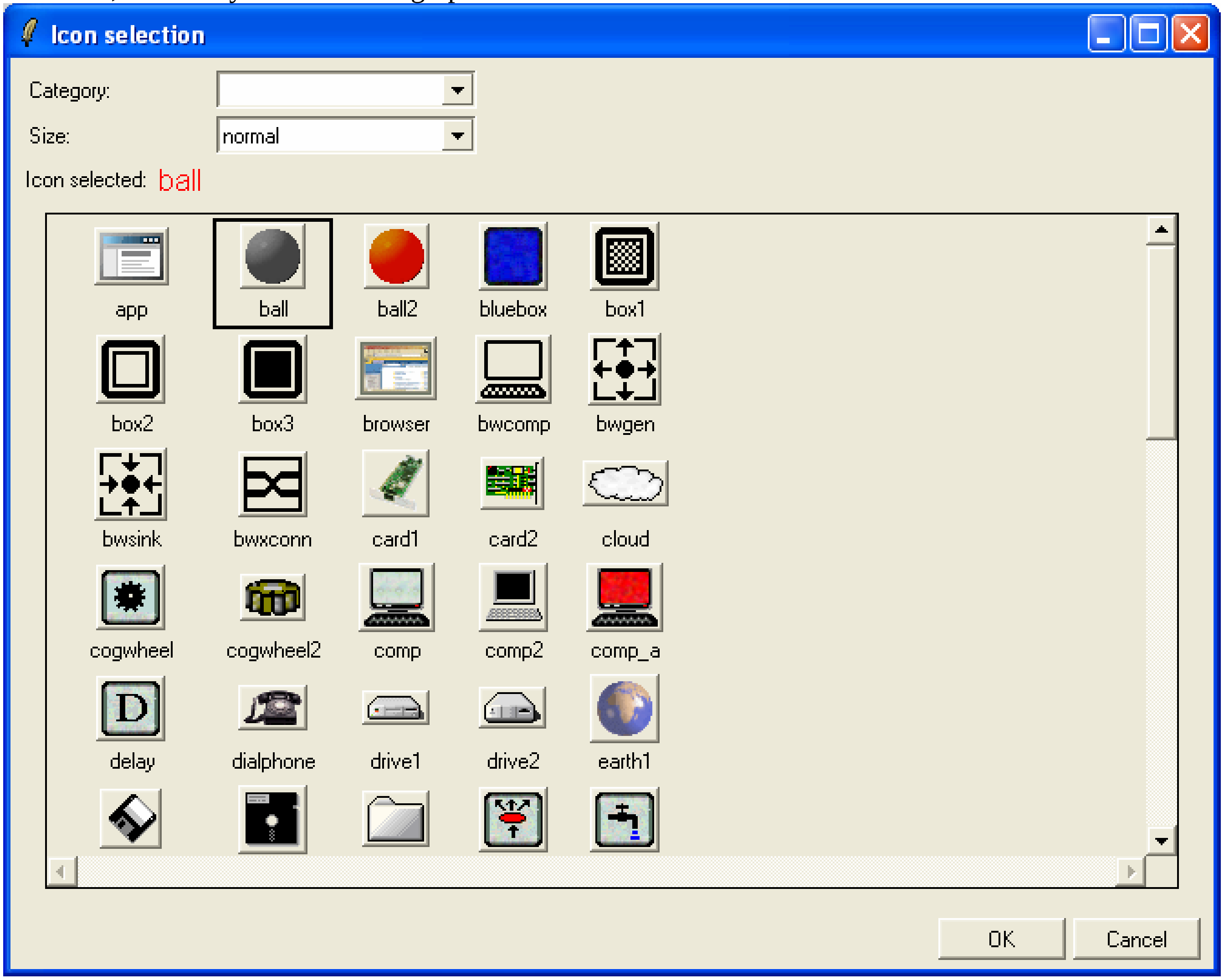
Line thickness: default

Fill Color:

Outline color:

Let us change the appearance of this module, to an icon representation. You can use one of the icons delivered with OMNEST^{TM}/OMNET++, or use any custom bitmap you have installed into the bitmaps directory of OMNEST^{TM}/OMNET++. Let us stick to a pre-installed icon for now.

Click the Icon radio button on towards the top of the dialog, and select the "ball" icon from the drop-down list. You could also have clicked on the "-n/a-" bellow the drop-down list, to select your icon via a graphical browser.



Icon selection

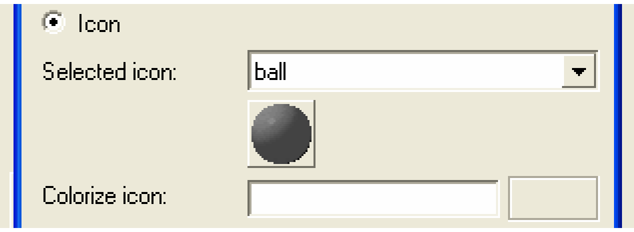
Category:

Size: normal

Icon selected: ball

Icons are available in four sizes, selectable through the drop-down list on top: large, normal, small, very small. Select "ball" from the "normal" size icons.

Now let us colorize this grey icon, to fit our taste. Let us make it blue!

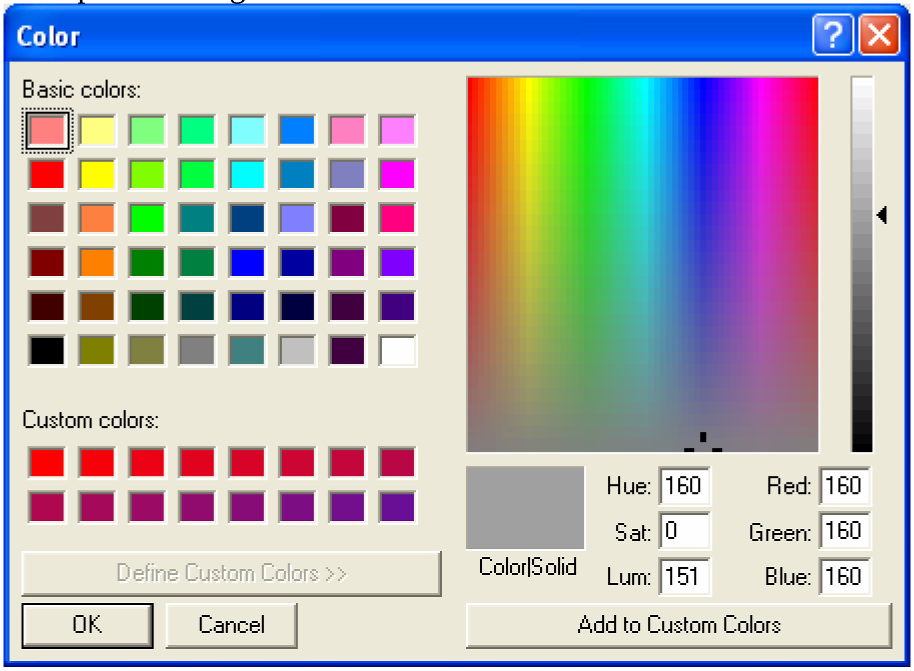


● Icon

Selected icon: ball

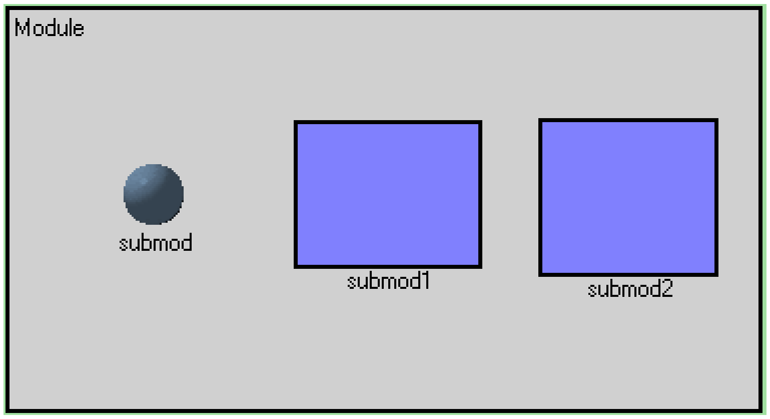
Colorize icon:

Click the grey rectangle next to the Colorize icon field. You will be presented with a color picker dialog.



Hue: 160
Sat: 0
Lum: 151
Red: 160
Green: 160
Blue: 160

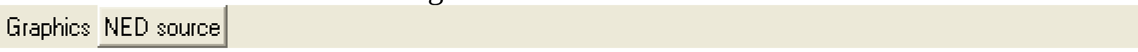
Select a blue color.



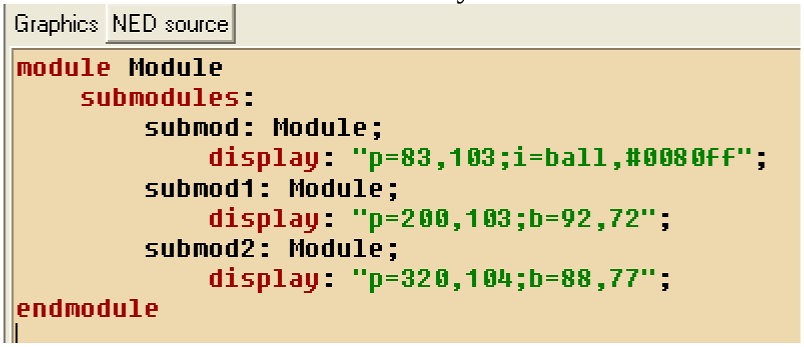
You will now see a blue ball instead of the rectangle you had previously, but for this submodule only.

Now let us see, how useful it is, that we can also edit in text mode! The change of icon, and colorization is straightforward and easy, but you might not want to repeat it for many submodules. Let us save time, by copying and pasting this information directly into the NED description of the other two modules, and let us also change their color at the same time.

Switch to NED source mode using the switch.

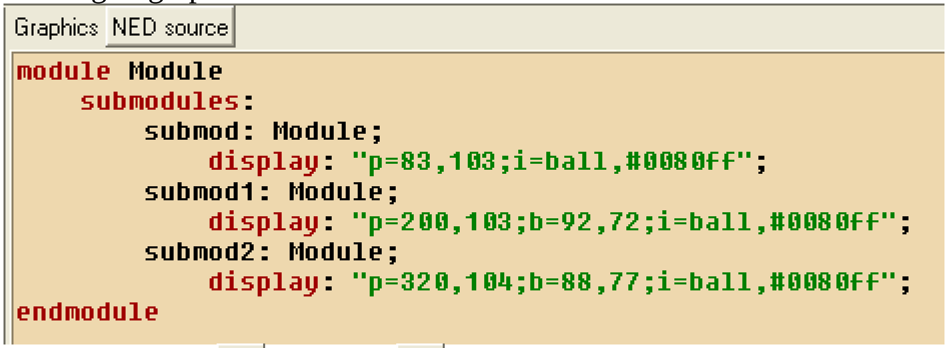


You should see the NED source of your model:



```
module Module
    submodules:
        submod: Module;
            display: "p=83,103;i=ball,#0080ff";
        submod1: Module;
            display: "p=200,103;b=92,72";
        submod2: Module;
            display: "p=320,104;b=88,77";
endmodule
```

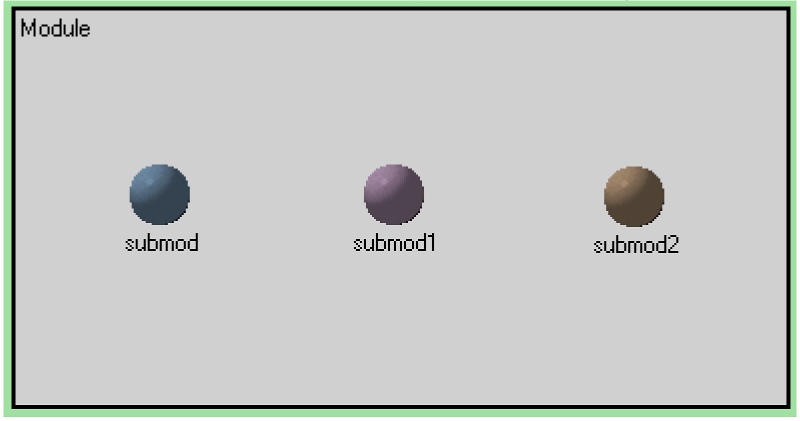
Notice the i=ball,#0080ff in the display string of “submod”. If we were to copy and paste the same text into the display string of “submod1” and “submod2”, we could save a lot of clicking in graphics mode.



```
module Module
    submodules:
        submod: Module;
            display: "p=83,103;i=ball,#0080ff";
        submod1: Module;
            display: "p=200,103;b=92,72;i=ball,#0080ff";
        submod2: Module;
            display: "p=320,104;b=88,77;i=ball,#0080ff";
endmodule
```

You can use the Copy and Paste buttons after selecting the text, or just use the familiar Ctrl+C and Ctrl+V keyboard shortcuts.

Now, let us make things a little more interesting. Change the colorization string (#0080ff), which is in fact an RGB value in “submod1” to #ff80ff and in “submod2” to #ff8000. Now click back to Graphics mode using the switch:

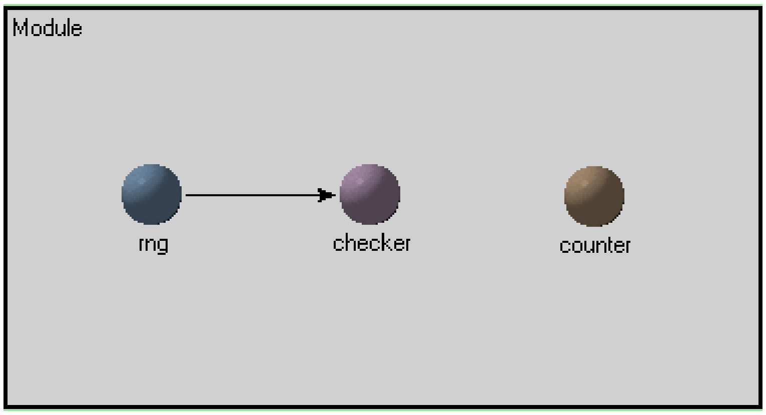


Notice, that GNED has kept up with all the changes you made in source mode, and it all reflects in the graphical representation of your model.

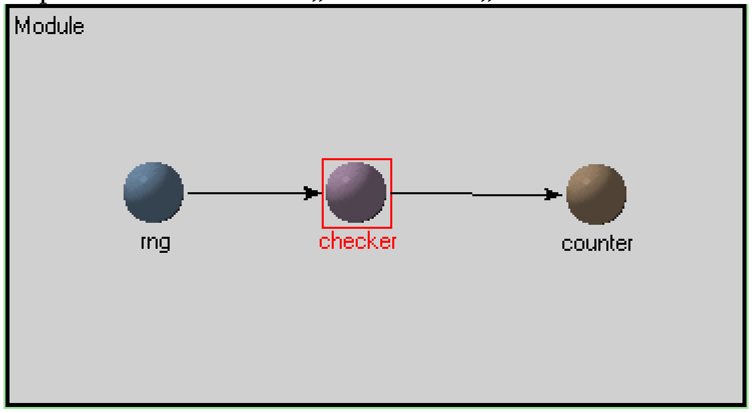
Now let us suppose that these "balls" represent three components of an abstract experiment, where we want to evaluate a specific property of three different Random Number Generators, or RNGs.
Rename the blue ball "rng", the lilac ball "checker", and the brown ball "counter". To rename a submodule, right-click on the ball, and select "Rename…" from the context sensitive menu.

"rng" will generate numbers, which it will turn over to "checker". "checker" will evaluate each number, and only send the even ones to "counter". Odd numbers will be thrown away. "counter" will record the number of even numbers it receives each minute, and send this number – when requested to an external component.

Let us create the necessary connections:
"rng" must connect to "checker", and "checker" has to connect to "counter". To connect two submodules, make sure, you are in "Draw submodules and connections" mode (the button is depressed). Click inside the blue "rng" ball, and drag into the lilac "counter" ball. GNED will create a connection between the two.

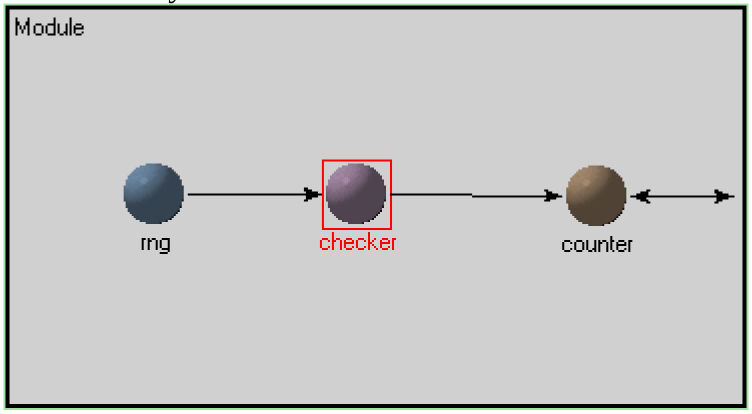


Repeat this action from „checker" to „counter".



Counter will also receive, and send messages to and from the outside world, so we should make the necessary connections.

To connect a module to the outside, drag from its center to the boundary of the grey area. To connect from the outside to the module, drag from the boundary to the center of the module.
This is what your result should look like:



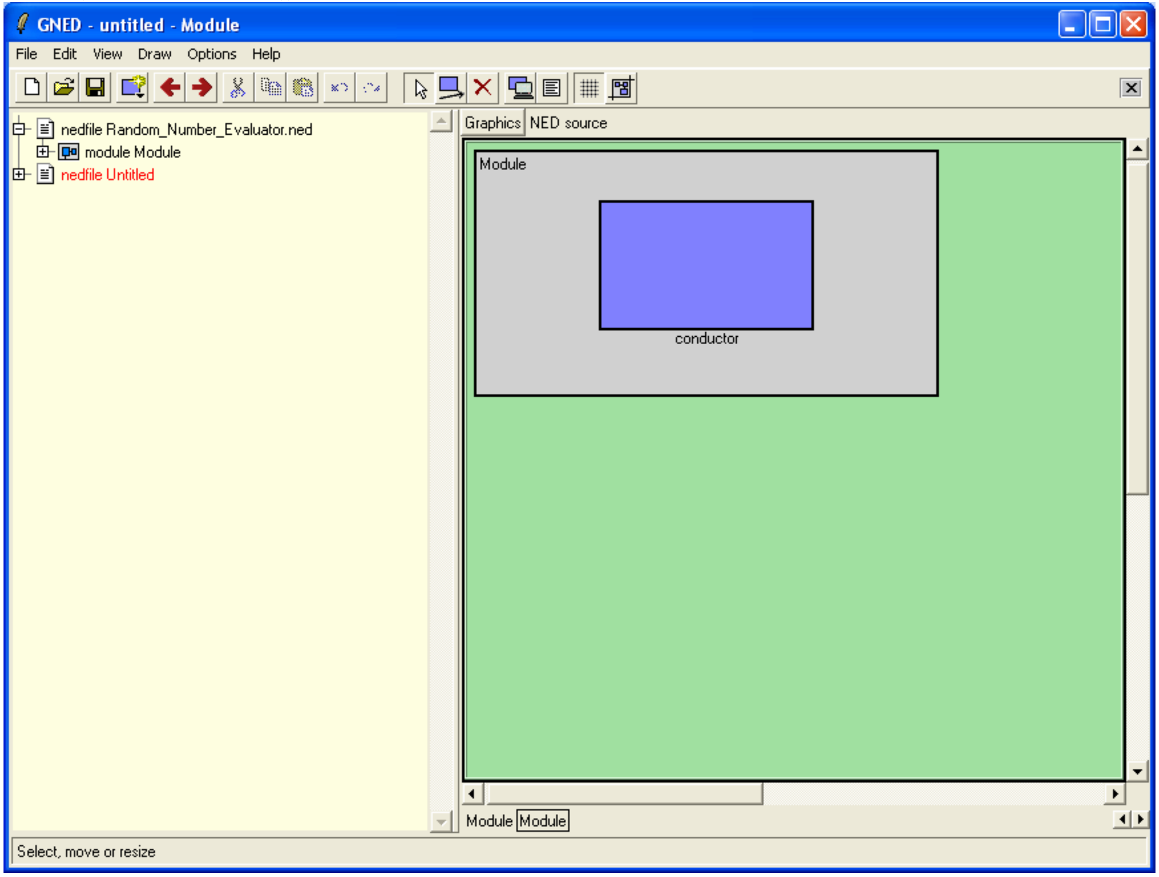
Now, to give this compound module a name, right-click on the “Module” text in the top left corner, and chose “Rename…” from the context menu. Let us call it ““rnd_eval””.

Let us also save the NED file containing the compound module, and call it “Random_Number_Evaluator.ned” use the “Save As…” command from the File menu, and save it to any convenient location.

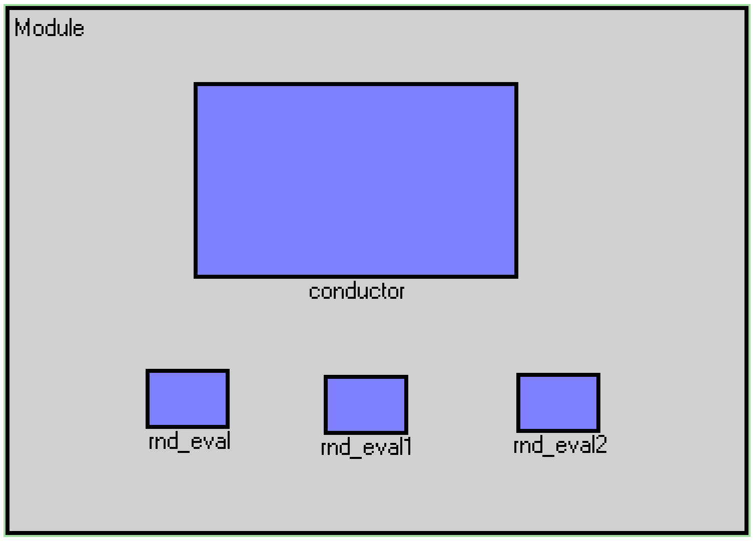
Now, let us create a new NED file, and make use of the compound module we just set up for an experiment.

3. Working with multiple NED files, advanced editing.

Press the 🗋 button to create a new NED document. Create a single submodule in the center, and rename it „conductor”.

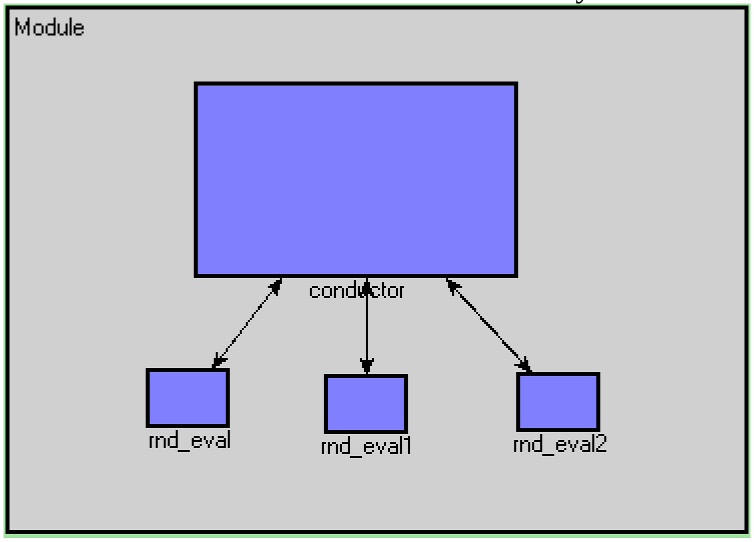


Now drag your „module rnd_eval” onto this canvas three times.

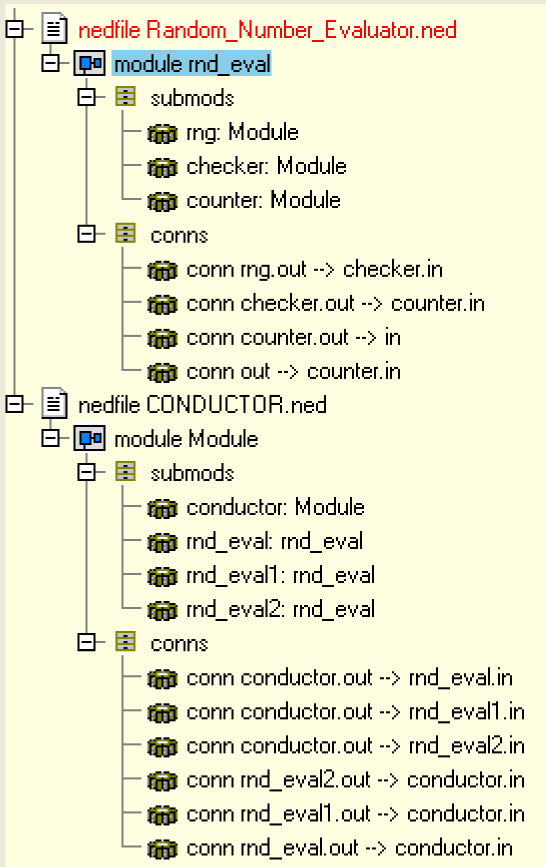


Notice, that GNED has automatically kept the name you have given your module, and extended it with an instance counter. Now create the connections between the “conductor” and the “rnd_eval” instances, making sure, they are two-way connections. The conductor will ask each “rnd_eval”from time to time, for their statistics, and they should answer back.

You must create each connection individually.

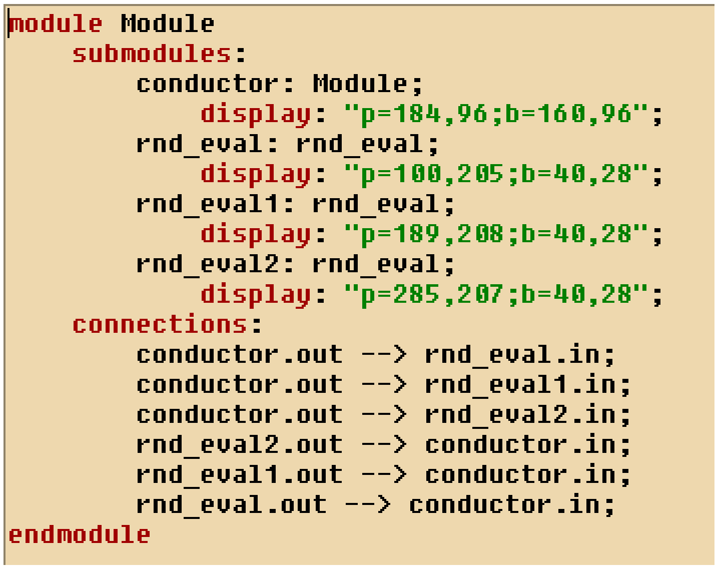


If you expand the tree view of the second NED file, you will see that these few “strokes” have resulted in quite an elaborate network:



```
nedfile Random_Number_Evaluator.ned
    module rnd_eval
        submods
            rng: Module
            checker: Module
            counter: Module
        conns
            conn rng.out --> checker.in
            conn checker.out --> counter.in
            conn counter.out --> in
            conn out --> counter.in
nedfile CONDUCTOR.ned
    module Module
        submods
            conductor: Module
            rnd_eval: rnd_eval
            rnd_eval1: rnd_eval
            rnd_eval2: rnd_eval
        conns
            conn conductor.out --> rnd_eval.in
            conn conductor.out --> rnd_eval1.in
            conn conductor.out --> rnd_eval2.in
            conn rnd_eval2.out --> conductor.in
            conn rnd_eval1.out --> conductor.in
            conn rnd_eval.out --> conductor.in
```

The NED source will tell the same tale:

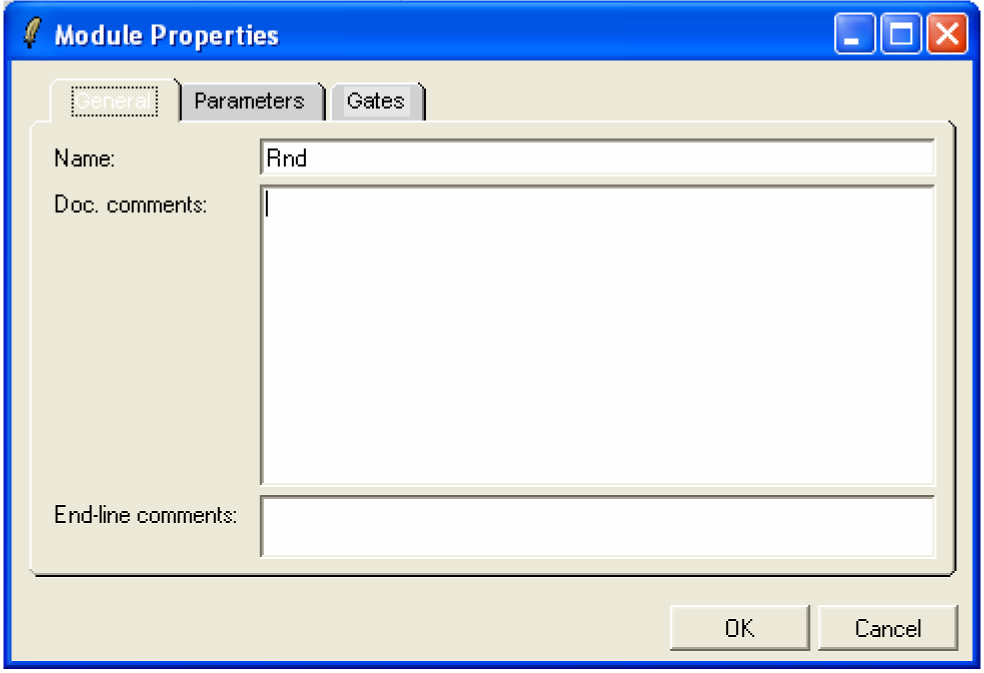


```
module Module
    submodules:
        conductor: Module;
            display: "p=184,96;b=160,96";
        rnd_eval: rnd_eval;
            display: "p=100,205;b=40,28";
        rnd_eval1: rnd_eval;
            display: "p=189,208;b=40,28";
        rnd_eval2: rnd_eval;
            display: "p=285,207;b=40,28";
    connections:
        conductor.out --> rnd_eval.in;
        conductor.out --> rnd_eval1.in;
        conductor.out --> rnd_eval2.in;
        rnd_eval2.out --> conductor.in;
        rnd_eval1.out --> conductor.in;
        rnd_eval.out --> conductor.in;
endmodule
```

Save this experiment under “Conductor.ned”.

For our example to be realistic, we need to define “Rnd”, “Checker” and “Counter” as simple modules, since these will all be implemented by their own C++ class.

As a first step, we must create three simple modules. Press the button, or use the context menu in the tree view to insert a new simple module. After reading the hint, you should see the properties dialog.

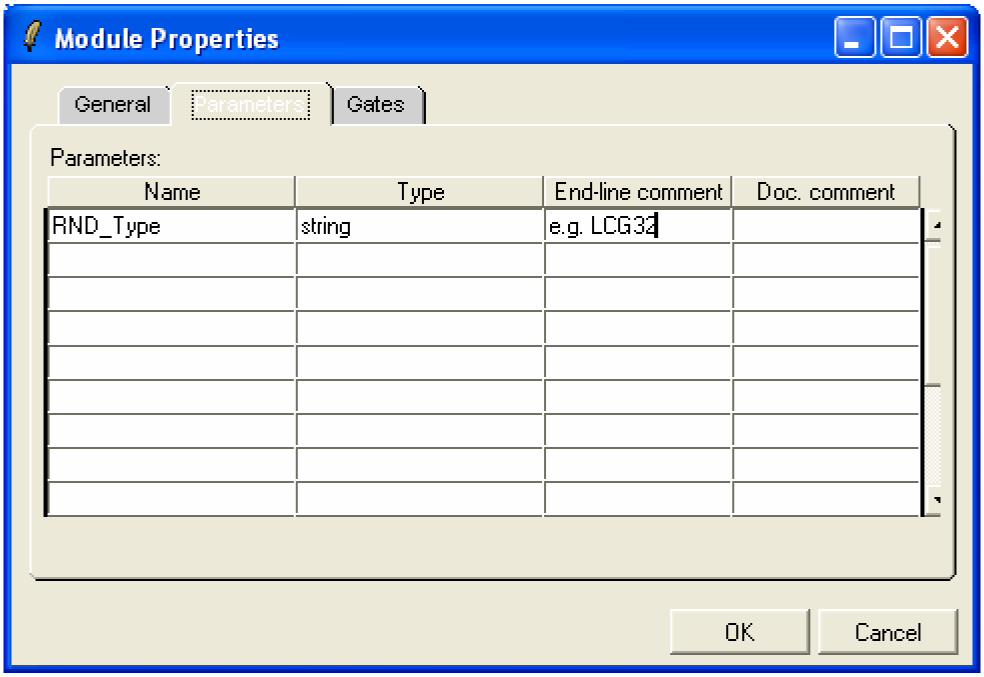


Name: Rnd

Doc. comments:

End-line comments:

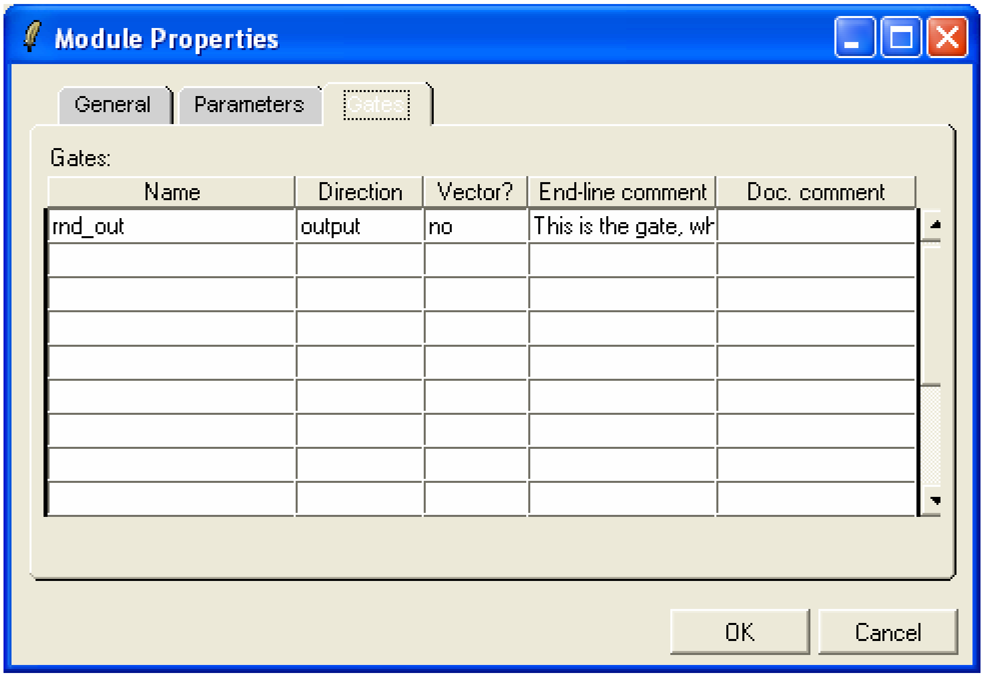
Select the “General” Tab and rename Simple to “Rng”.



Parameters:

Name	Type	End-line comment	Doc. comment
RND_Type	string	e.g. LCG32	

Click the “Parameters” tab, and define the parameter, by which you can later externally configure, which RNG algorithm should be used.



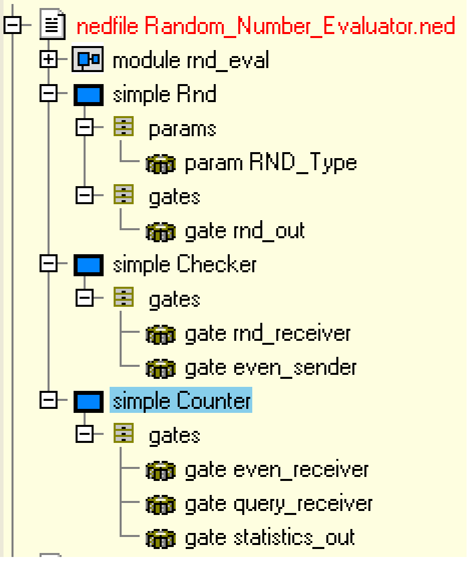
Gates:

Name	Direction	Vector?	End-line comment	Doc. comment
rnd_out	output	no	This is the gate, wh	

Click the “Gates” tab and define a gate, through which the generated random numbers will be passed along. You can comment the object, with something like “This is the gate, where the random numbers are passed along.”

The Rnd simple module is now ready to be used by Rnd.cc and Rnd.h, which will have to be implemented according to other sections of the manual.

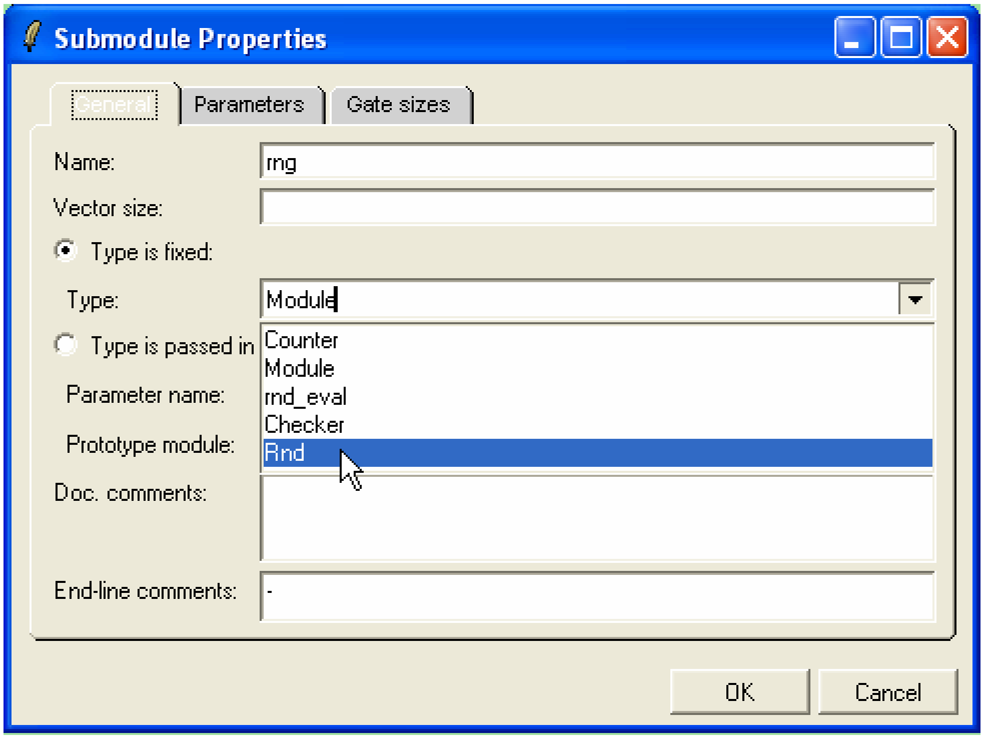
You need to create simple modules for “checker” and “counter” as well, and define their parameters and gates in similar fashion. You should have something like this:



```
nedfile Random_Number_Evaluator.ned
  module rnd_eval
  simple Rnd
    params
      param RND_Type
    gates
      gate rnd_out
  simple Checker
    gates
      gate rnd_receiver
      gate even_sender
  simple Counter
    gates
      gate even_receiver
      gate query_receiver
      gate statistics_out
```

Now change the type of each module inside rnd_eval in Random_Number_Evaluator.ned to its corresponding simple module type.

Right-click on “rng”, and select “Properties…” Then in the Type drop-down list, select Rnd.



Name: rng

Vector size:

● Type is fixed:

Type: Module

○ Type is passed in

Parameter name:

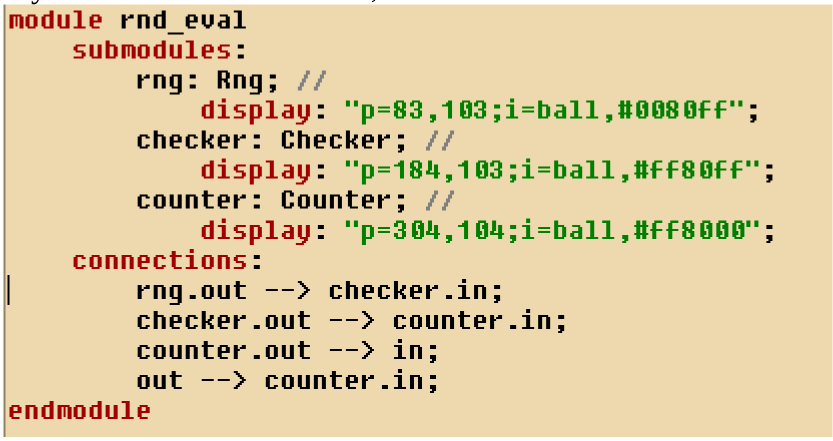
Prototype module:

Doc. comments:

End-line comments: -

Modify the type of “checker” and “counter” in similar fashion, and save the files.

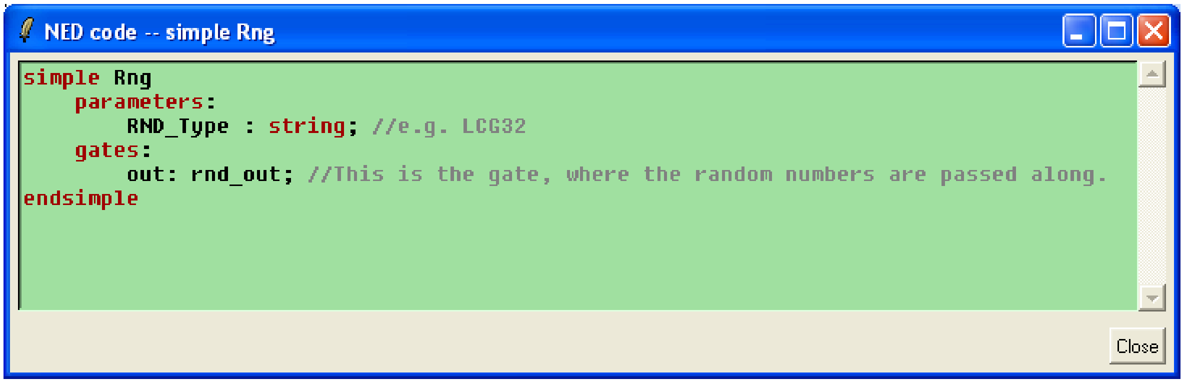
If you check the source view,



```
module rnd_eval
    submodules:
        rng: Rng; //
            display: "p=83,103;i=ball,#0080ff";
        checker: Checker; //
            display: "p=184,103;i=ball,#ff80ff";
        counter: Counter; //
            display: "p=304,104;i=ball,#ff8000";
    connections:
        rng.out --> checker.in;
        checker.out --> counter.in;
        counter.out --> in;
        out --> counter.in;
endmodule
```

you will see, that rng, “checker” and “counter” are no longer of a generic “Module” type, but have their own simple module definitions.

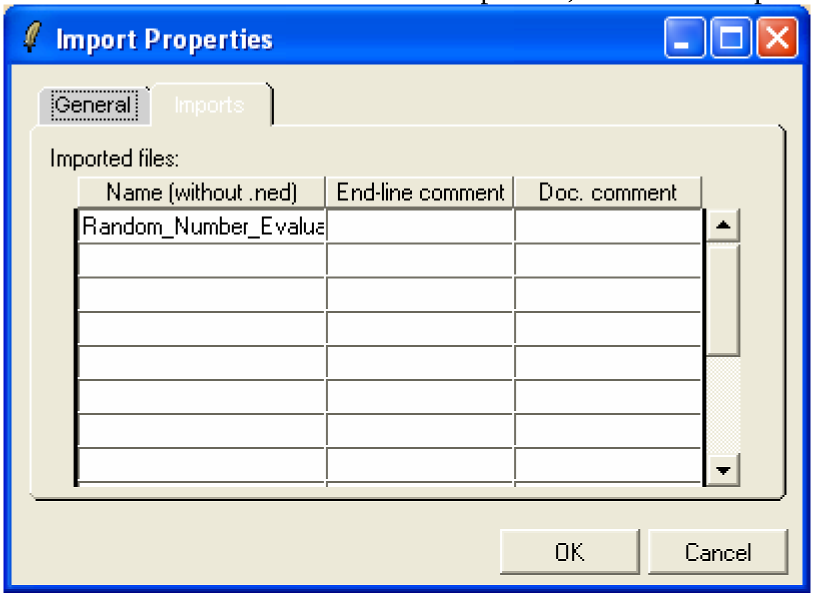
Also, by accessing the context menu, (right click) on the simple module “Rng” and pressing “Show NED code…”, you will see, that your settings in the “Parameters” dialog have also resulted in NED code:



```
simple Rng
    parameters:
        RND_Type : string; //e.g. LCG32
    gates:
        out: rnd_out; //This is the gate, where the random numbers are passed along.
endsimple
```

OK, so what is missing from the Conductor.ned file? Well, we are using “rnd_eval”, a compound module that is defined in a different NED file, than Conductor.ned, so we should list it as an import in that file, to be able to make use of it.

Use the button to add a new component, and chose Imports.



Imported files:

Name (without .ned)	End-line comment	Doc. comment
Random_Number_Evalua		

Enter „Random_Number_Evaluator” into the Name column, and add any comments you might want in the next two columns. These comments are used by the automatic documentation tool; please see the manual for details.

If you now save your conductor.NED file, you have finished putting together a simple network.

To create a simulation you can run, you have to implement the simple modules in C++, and build an executable. The Manual provides more information about this.

This completes your basic introduction to the GNED editor. Enjoy!