Sphinx SD Tools User Manual

Sphinx SD Tools Developers Team
R.G. Zaytsev P.G. Kuzmenko I.V. Maltsev

April 2, 2008

Contents

Chapter 1

Introduction to System Dynamics

1.1 System Dynamics

Under development

1.1.1 System Thinking

Under development

1.1.2 Causal Loop Diagram

Under development

1.2 A Modelling Approach

Under development

1.2.1 Basic Structures

Under development

1.2.2 Feedback Structures

Under development

Chapter 2

System Dynamics modelling with Sphinx SD Tools

2.1 What is Sphinx SD Tools?

Sphinx SD Tools is an modelling environment that allow you create system dynamics models, simulate them and view results of the simulation. Main features that Sphinx SD Tools offers you are:

- Graphical System Dynamic Model Editor
- Outline view of creating model
- Textual System Dynamic Model Editor
- System Dynamics Model Templates
- Enhanced Formula Editor
- Model simulation
- Chart and Table views of System Dynamic Model simulation results
- Multi-project approach

Multi-project approach

Sphinx SD Tools allow you to edit and simulate different models at the same time. Using `File` menu as shown at Figure 2.1 you can create or open multiple projects. Project in terms of Sphinx SD Tools aggregates one System Dynamics model, it's simulation settings, possibly multiple charts and tables

for showing simulation results with their own settings. At Figure 2.2 you can see an example of two models being modelled and executed at the same time.

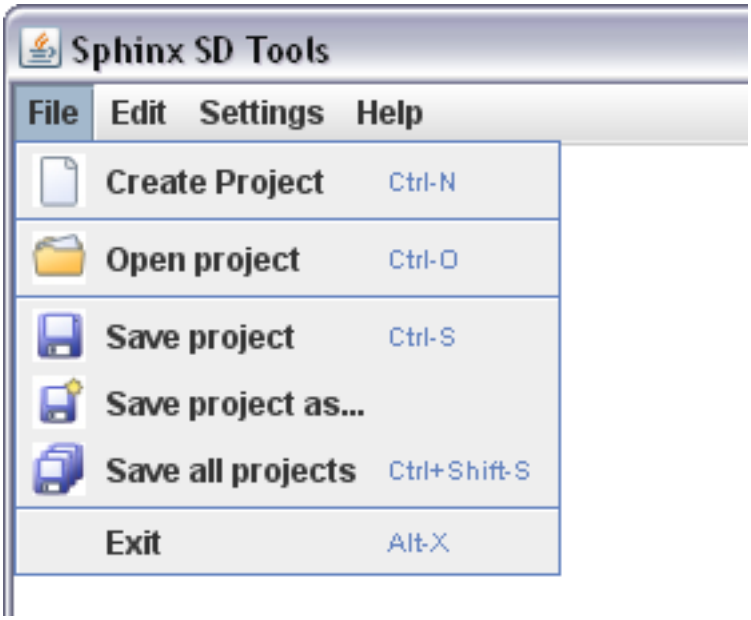


Figure 2.1: File menu

Once you've created or opened project you are able to edit System Dynamics model. There are two ways to edit model:

- Graphical editor
- Text editor

2.2 Editing model with Graphical Editor

2.2.1 Adding elements to model

Graphical Editor provides more easy to use model editing approach than text editor does. At Figure 2.3 you can see Graphical Editor for newly created model.

At the left side you can see tool bar that offers you instruments for creating model. You can click and drag them to the editor field:

-  `Stock` to add a stock element to model

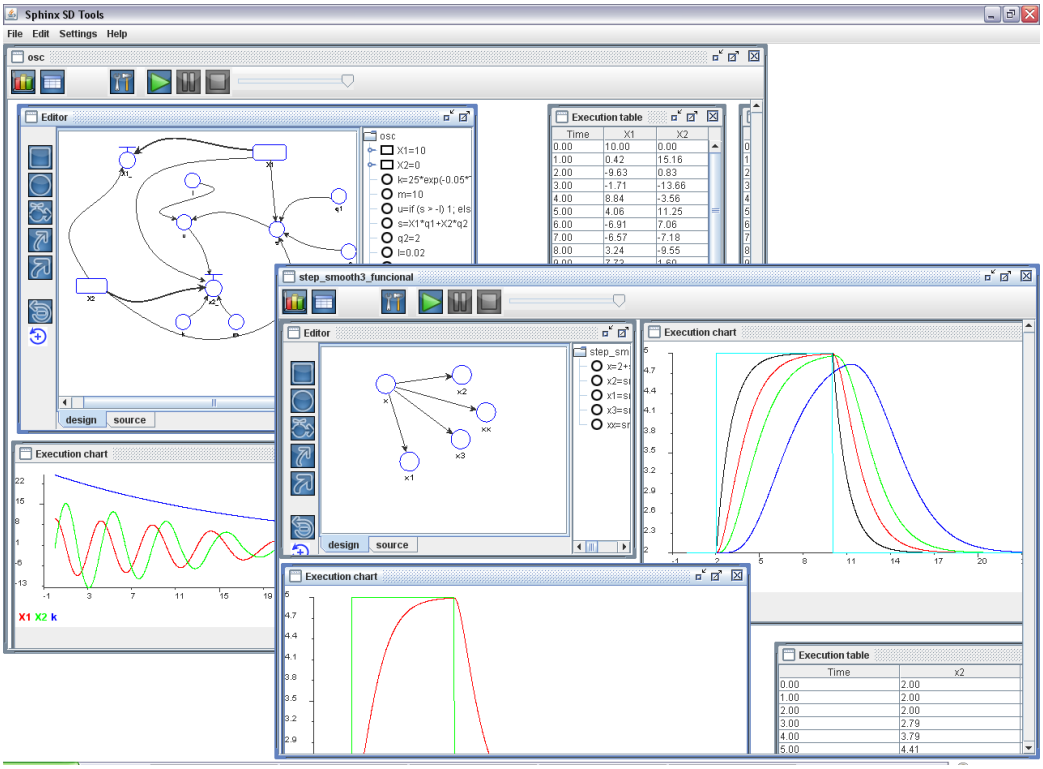


Figure 2.2: Two models at the same time

-  `Converter` to add a converter or constant element to model
-  `Flow` to add a flow element to model that initially not connected to any other elements

After you've added an element at the editor's field you can rename it by double-clicking on it.

2.2.2 Adding links to model

Below that instruments you can find two icons with arrows. To connect two elements at the model you should first click on the desirable icon, than click on the element from which arrow should begins and drag it to element at which arrow should have ending. Meanings of the arrows are:

-  `Action connector` is an informational link between two elements

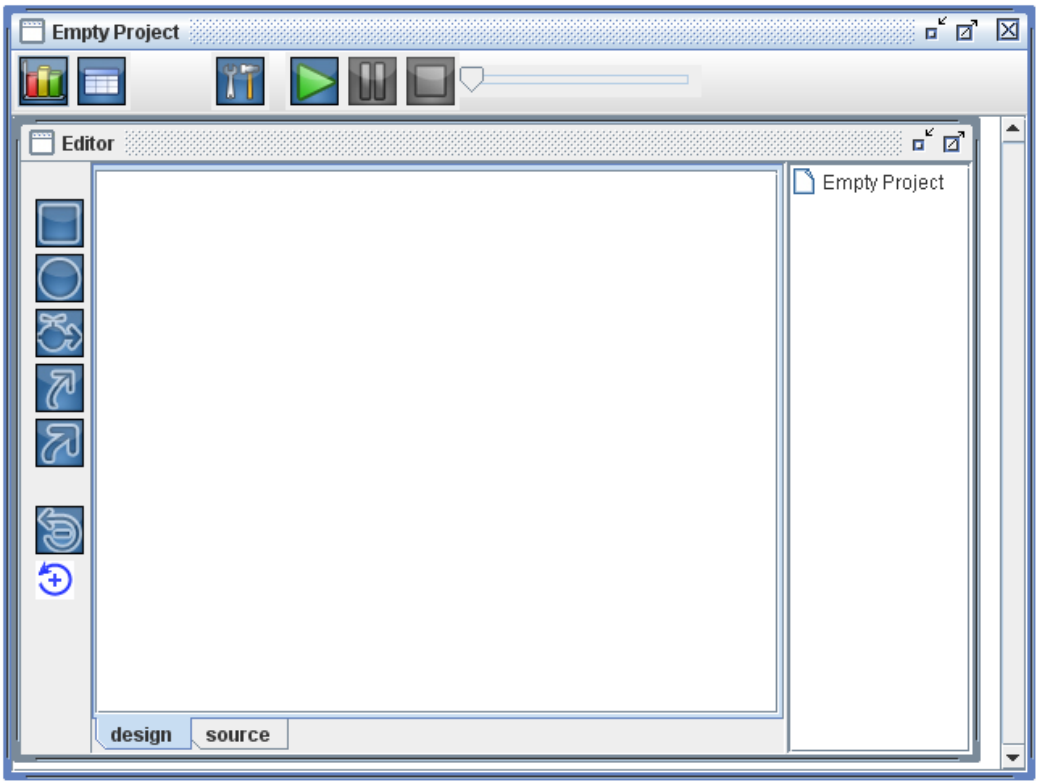


Figure 2.3: Empty System Dynamics model Graphical editor

-  `Stream` to create a stream from `Stock` to `Flow` or vise versa.

If you want links to be not just straight you should add one ore more "control points" between it ends (Figure 2.4). You may done that by clicking at the line by right mouse button while Left Shift button is pressed. After that you may drag additional point and make such a curve as you wish. If you want to remove additional "control point" than click by it again with right mouse button while holding Left Shift down.

2.2.3 Adding templates to model

And, finally, below described instruments you may find some instruments called `Templates`. Templates are set of linked elements with predefined dependencies. For example, we have an "First-order Negative Feedback Loop" template (Figure 2.5). You can drag templates like `Stocks`, `Flows` or `Converters` and drop them at the editor field of the editor to add them to the model being edited.

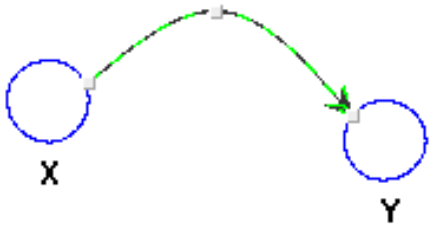


Figure 2.4: Making Curve Line

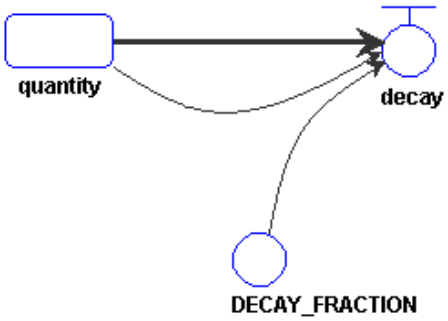


Figure 2.5: First-order Negative Feedback Loop Template

2.2.4 Modifying model

At the editor field you can perform following operations:

- Select one or more elements by clicking at one element or dragging mouse while it's left key is pressed (marquee tool)
- Move selected elements
- Delete selected elements
- Copy/Paste parts of model
- Zoom In/Out using mouse wheel
- Access described operations and some element-specific operations with pop-up menu that is available through right mouse click as you can see at Figure 2.6 and Figure 2.7.

As you may notice, there is some special item in pop-up menu for element `Stock`: check-box `Non negative`. If checked, it informs that at simulation time associated `Stock`'s value cannot fall beyond 0.

2.2.5 Formula Editor

Formula editors allows you to easily edit formulas for elements of System Dynamics model with formulas validation for syntactic and semantic correctness. At Figure 2.8 you can see formula editor that you can open by clicking at the `Formula Editor` at the pop-up menu of element (Figure 2.6)

Formula editor overview

As you can see, editor dialog is virtually divided into 4 parts:

Formula Input Field This is main part of formula editor. Here you can enter desirable formula from your keyboard or, possibly, paste from clipboard.

List of Inputs This list contains all the elements that are input elements for element being edited in model. List of inputs may be of two types:

- List of Available inputs — the list of input that you CAN use in formula as variable but NOT MUST.

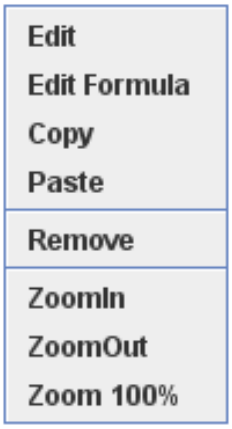


Figure 2.6: Pop-up menu for element

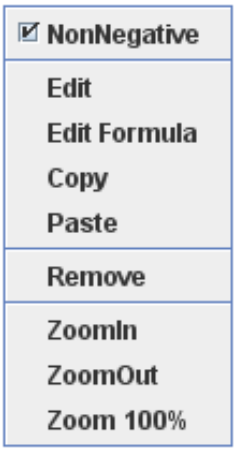


Figure 2.7: Pop-up menu for Stock element

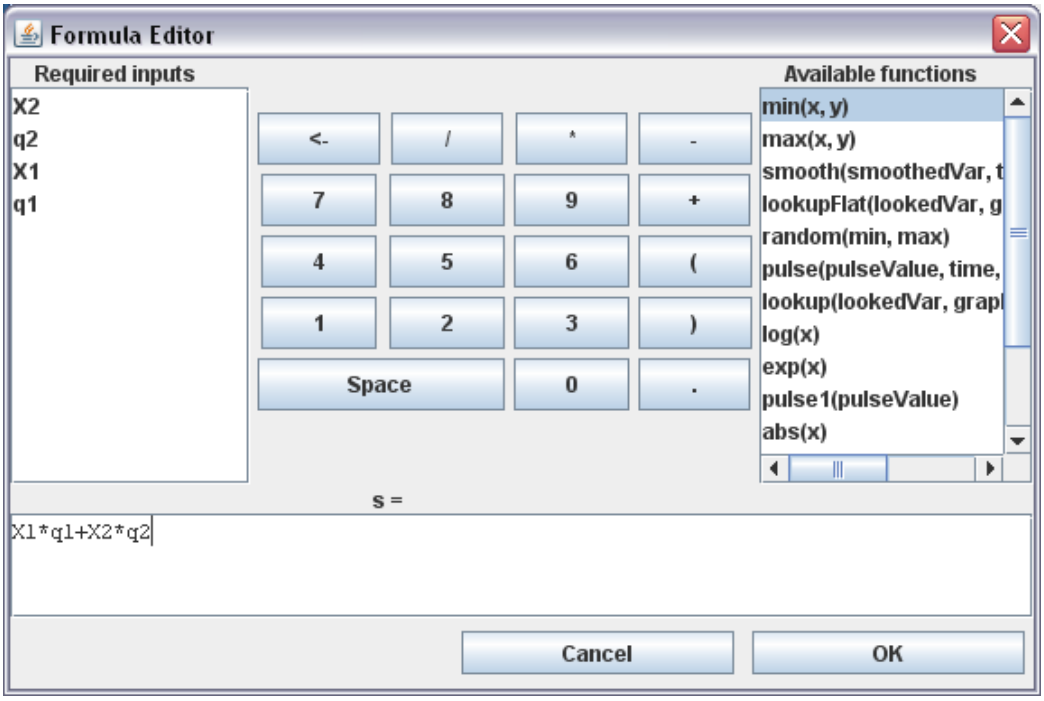


Figure 2.8: Formula editor dialog

- List of Required inputs – the list of inputs that MUST be used in formula, moreover, ALL of them must be used in formula.

Clicking at the name of element in the List of Inputs will insert it name in the Formula Input Field at the cursor position.

Virtual Keyboard This is a small virtual keyboard that you can use to type in characters into Formula Input Field with your mouse. Clicking at the appropriate button will insert associated character in the Formula Input Field at the cursor position.

Function List This is list of available functions you can use while building your formula. Clicking at the function name in the Function List will insert template of function with it's parameters into the Formula Input Field at the cursor position. For example, if you wish to add function that takes maximum from two parameters you may click at the `max(x, y)` function and than `max( , )` will be inserted.

Later, in the **??** we will describe all the functions you able to use and describe their behaviour.

Once you've entered formula you can press `OK` to accept formula or reject it by pressing `Cancel` button.

Formula typing

Formal specification of formula syntax may be found in Chapter 3.2 at page 18.

But in less formal way we can describe formula expressions syntax as a subset of `JavaScript` language for typing mathematical expressions with extension to have an ability to specify table parameters (for "lookup functions").

So, we will give you some valid and invalid examples of formula expressions. The following formulas will be accepted:

- -2 — any integer, both positive and negative.
- x, $\_a$, $y12$, $\_4$ — any valid variable names. Variable name can starts with '_' or any character from range $[a..z]$ both in upper or lower cases. Following characters may be mixed with digits.
- $-.0$, 0.123 — any float-point, both positive and negative.
- $a()$,
  $a(b(), -c)$ — any function calls with valid parameters.
- $if\ (a > -b)\ c;\ else\ d;$,
  $if\ (a > -b)\ c;\ else\ if\ (d < e)\ f;\ else\ \{if\ (g < h)\ \{i;\}\ else\ \{j;\}\}$
  — any `if`-statements with their `then` and `else` clauses ending with ';'. Additionally, `then` and `else` clauses may be surrounded by { and } braces. Both `then` and `else` clauses are required.
- $lookup(x, [(1, 2)]$,
  $lookupFlat(y, [(1, 2), (1.2, -3)]))$ — any function calls with table parameters (also known as table functions, lookup functions and so on)[1].

The following formulas will be rejected:

- $12df$ — function name or variable cannot starts from digit.
- $a(s + d()$ — forgotten enclosing brace.

[1]Please note that for the time of writing the only functions that can accept table parameters (in other words, table parameters usage semantically correct) are `lookup` and `lookupFlat`. They should be used as pointed above

- $if\ (a > b)\ a\ else\ c$ — forgotten ';' at the ends of `then` and `else` clauses. Should be $if\ (a > b)\ a;\ else\ c;$.
- $if\ (c > d)\ f;$ — missed `else` clause.
- $lookup(x, [])$ — table parameter should contain at least one pair of numbers.
- $lookup(y, [(1, 2), (3, 4),])$ — redundant ','.

Error handling

If you pressed `OK` button but formula contain errors than error message dialog will appear as shown on Figure 2.9. When you see such a dialog you have

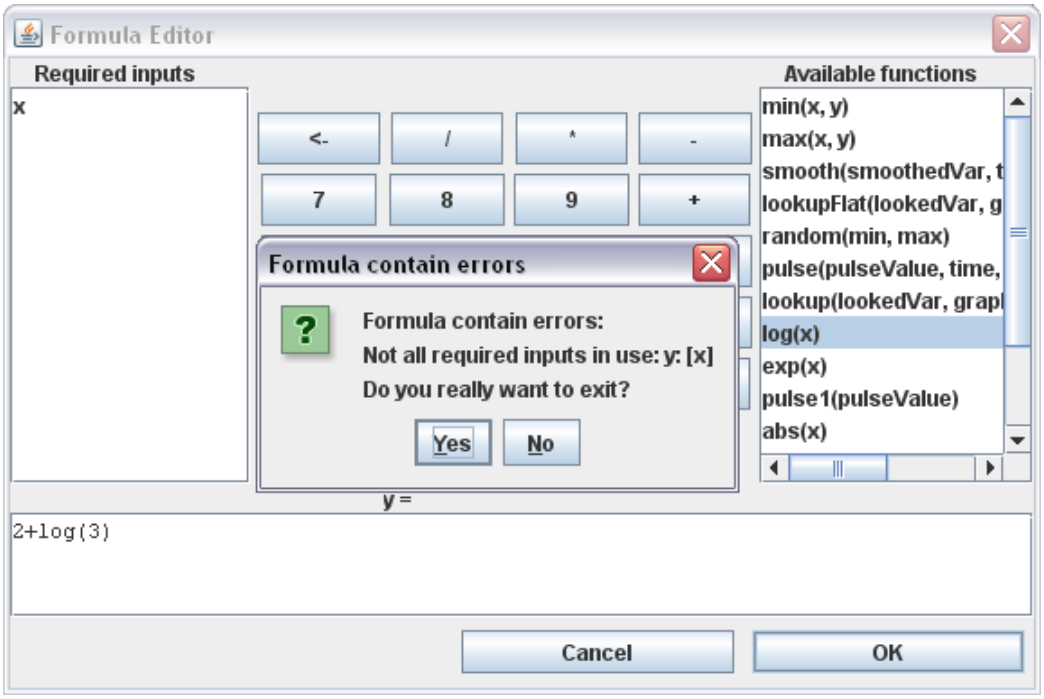


Figure 2.9: Error in formula: Not all required inputs are in use

two options:

- Press `Yes` and exit without saving changes you made to formula for element
- Press `No` to get back to formula editor and correct your mistake

There two kind of errors:

- Syntactic errors, errors that appeared if you entered formula with invalid syntax. An example of such error you can see on Figure 2.9 where you entered incorrect formula `2+x+`
- Semantic errors, errors that appeared during illegal use of functions and variables (Figure 2.9)

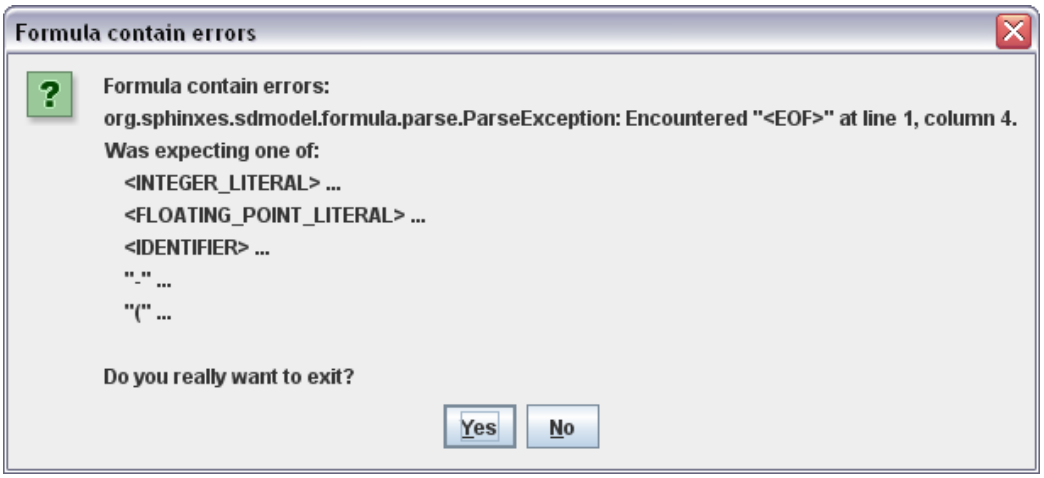


Figure 2.10: Error in formula: invalid syntax

2.3 Simulating model

The purpose of creation of any System Dynamics model is to simulate it over the some period of time and watch it behavior. In this chapter we will discuss how you can simulate models created using Sphinx SD Tools environment.

2.3.1 Simulating System Dynamics models

Under development

2.3.2 Simulation settings

You can set simulation settings that is specific for individual project. They are saved and loaded together with model within project.

To enter/edit simulation settings for a project you can press button  and you will see dialog shown on Figure 2.11. In this dialog you can set up following simulation parameters:

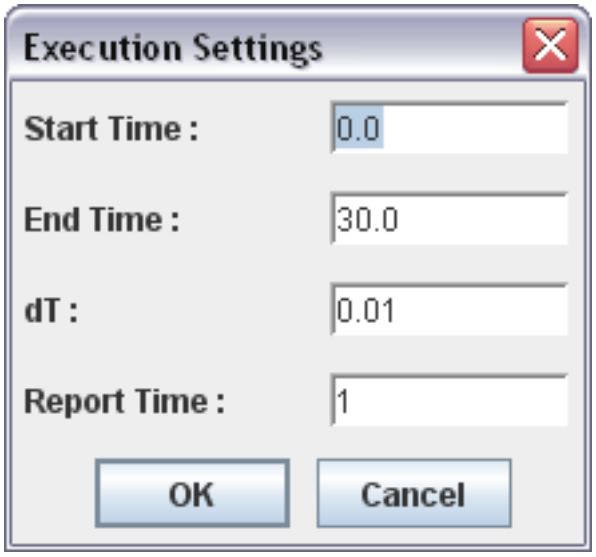


Figure 2.11: Model Simulation Settings Dialog

Start Time This is an initial value for variable `TIME` for model simulation.

End Time This is a final value for variable `TIME`. When `TIME` value will become greater then `End Time`, simulation will be stopped.

dT This is a value for Δt which is used for increment `TIME` value at the simulation time.

Report Time This is a period of time with which state of the model's variables is propagated to `Table view` (see 2.3.4).

2.3.3 Controlling simulation

After you've setted-up simulation settings you are ready to start simulation. You can control simulation with following buttons:

-  `Start` button, used to start/resume simulation.
-  `Pause` button, used to pause simulation, later may be resumed by `Start` button.
-  `Stop` button, used to stop simulation.

Please note that in both started and paused states of model simulation you cannot edit model. Model can be edited only in stopped state.

Errors during simulation

There two kind of errors that can occur when you want model to be simulated:

- Model simulation cannot be started because of model is incomplete. This may be because you've not to enter some formulas, you loaded project, but formulas is incorrect, incorrect names is used in model and so on. You can see an example of such a model at Figure 2.12.

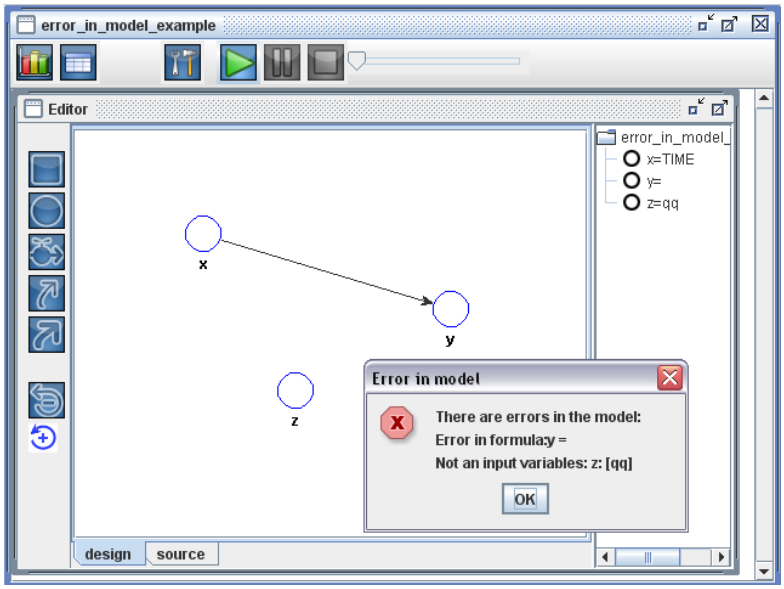


Figure 2.12: An example of incomplete model being started for simulation

- During model simulation occur computational error that leads some element's value to a `Not-a-Number` value (`NaN`) or to an `Infinity` value. For example, this can happen if you will try to calculate *log* function from negative argument ($log(-10)$) or divide something by zero ($10/0$) respectively. In such a case you will be prompted about error with error dialog as shown on Figure 2.13 and simulation will be stopped.

2.3.4 View simulation results

System Dynamics model simulation itself means nothing without results of simulation process. Sphins SD Tools offers you two tools for viewing results of the simulation process:

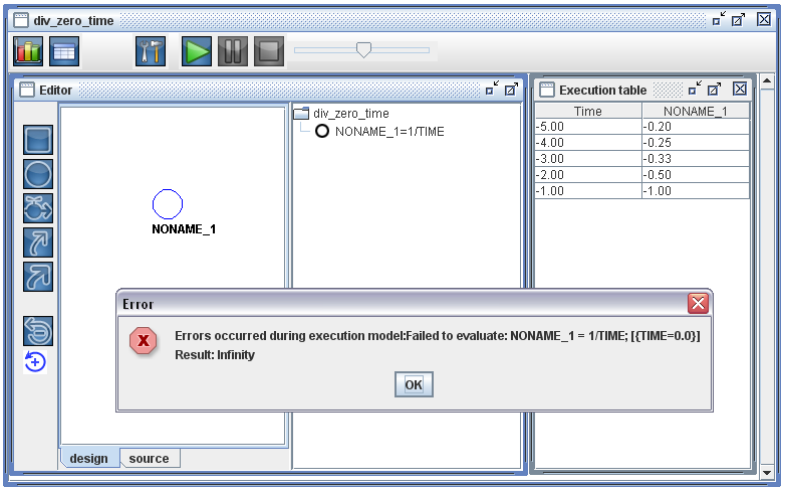
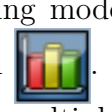


Figure 2.13: Division by zero occurred during model simulation

- Chart view
- Table view

Chart view

At Chart view you can visually watch the process of element's value changing during model simulation. To add Chart view to you project just click at icon  . Empty Chart view window will appear (Figure 2.14). You can add multiple Chart views to your project.

By clicking at `Model settings` button in Chart view you can access to the settings as shown on Figure 2.15.

At `Model entities` tab you have two lists:

Available List of all entities in model.

Selected List of entities selected to be shown on the chart.

You can move model's entities from one list to another by clicking at buttons:

- `>` to place selected element from Available list to Selected list.
- `>>` to place all elements from Available list to Selected list.
- `<<` to remove all elements from Selected list.

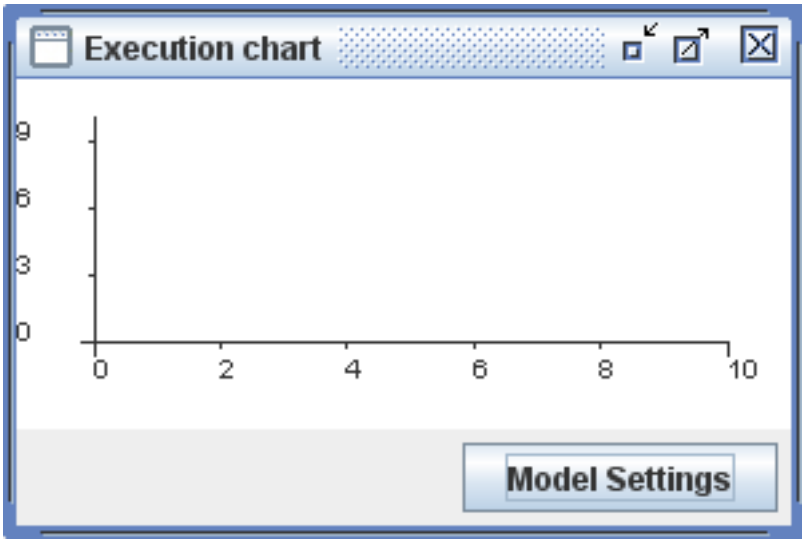


Figure 2.14: Empty Chart view

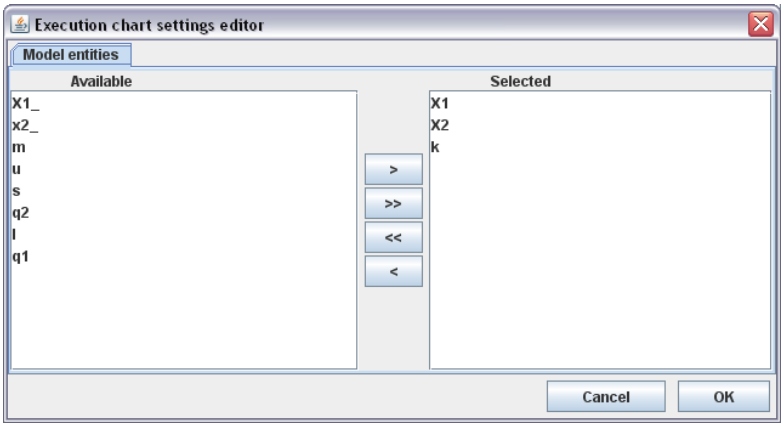


Figure 2.15: An example of Chart view settings dialog

- < to remove selected element from Selected list.

Click OK and you've done creating setting up settings view.

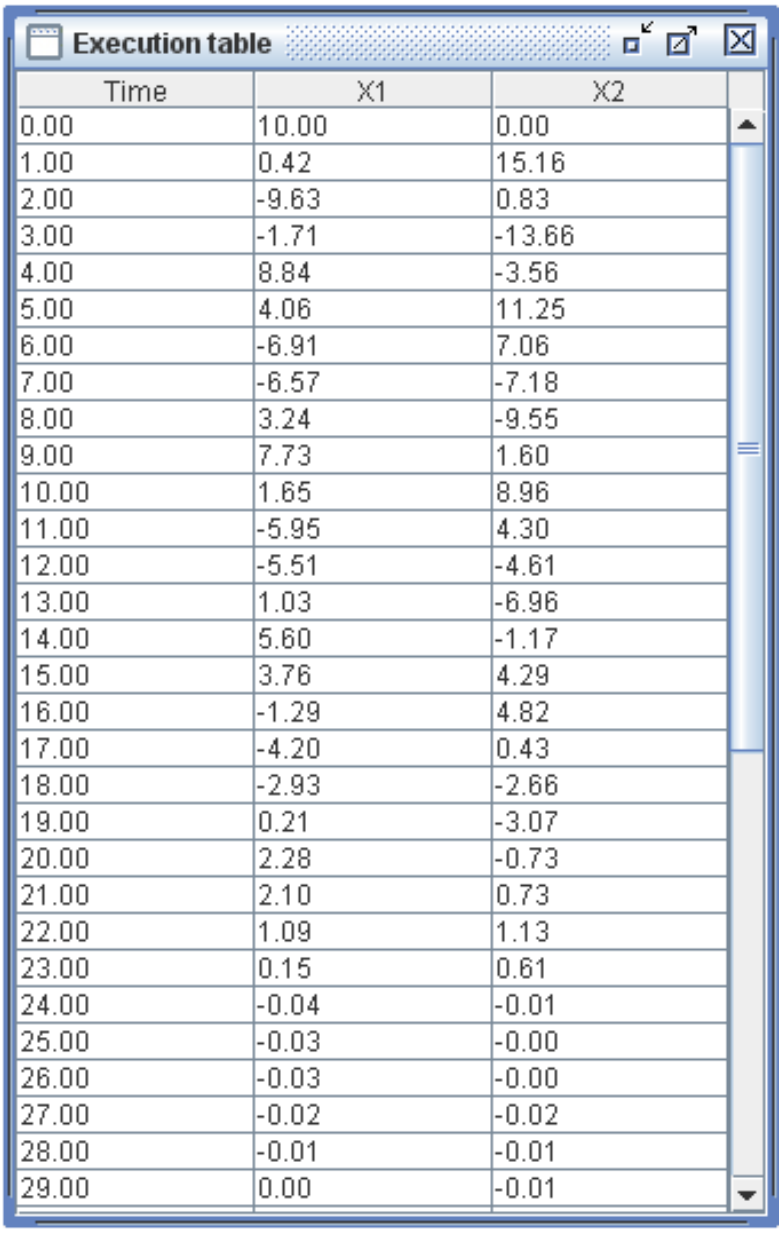
At the simulation time Chart view will automatically be stretched to show full chart of selected entities. Also, you have an ability to Zoom In by selecting with your mouse area you want to be zoomed and an ability to Zoom Out available through pop-up menu of the Chart view.

Table view

At Table view you can watch the process of element's value changing during model simulation at the table. To add Table view to you project just click at icon  . A Table view have much in common with Chart view, so can add multiple Table views to your project, edit settings that is the same as for Chart view (but available through pop-up menu). Example shown on Figure 2.16

2.3.5 Simulation methods

Under development



Time	X1	X2
0.00	10.00	0.00
1.00	0.42	15.16
2.00	-9.63	0.83
3.00	-1.71	-13.66
4.00	8.84	-3.56
5.00	4.06	11.25
6.00	-6.91	7.06
7.00	-6.57	-7.18
8.00	3.24	-9.55
9.00	7.73	1.60
10.00	1.65	8.96
11.00	-5.95	4.30
12.00	-5.51	-4.61
13.00	1.03	-6.96
14.00	5.60	-1.17
15.00	3.76	4.29
16.00	-1.29	4.82
17.00	-4.20	0.43
18.00	-2.93	-2.66
19.00	0.21	-3.07
20.00	2.28	-0.73
21.00	2.10	0.73
22.00	1.09	1.13
23.00	0.15	0.61
24.00	-0.04	-0.01
25.00	-0.03	-0.00
26.00	-0.03	-0.00
27.00	-0.02	-0.02
28.00	-0.01	-0.01
29.00	0.00	-0.01

Figure 2.16: Table view filled with data of model simulation's result

Chapter 3

Advanced Topics

In this chapter we will discuss some features that you usually will not need to use but in some cases may be found useful.

3.1 Editing model with Textual Editor

Under development

3.2 Formula syntax

Here you will find syntax of formulas used in Sphinx SD Tools in System Dynamics models. This is `BNF` notation that was generated by `JavaCC` tool.

```
NON-TERMINALS

formula ::= ( MathExpression | IfStatement ) <EOF>

MathExpression ::= NonNegativeMathExpression

NonNegativeMathExpression ::= InclusiveOrExpression

InclusiveOrExpression ::= ExclusiveOrExpression (
        "|" ExclusiveOrExpression )*

ExclusiveOrExpression ::= AndExpression (
        "^" AndExpression )*

AndExpression ::= AdditiveExpression (
```

```
        "&" AdditiveExpression )*

AdditiveExpression ::= MultiplicativeExpression (
        "+" MultiplicativeExpression |
        "-" MultiplicativeExpression )*

MultiplicativeExpression ::= UnaryMathExpression (
        "*" UnaryMathExpression |
        "/" UnaryMathExpression |
        "%" UnaryMathExpression )*

UnaryMathExpression ::=
        UnaryNonNegativeMathExpression |
        "-" UnaryNonNegativeMathExpression

UnaryNonNegativeMathExpression ::= FunctionCall |
        Literal |
        Id |
        "(" MathExpression ")"

FunctionCall ::= FunctionName
        "(" FunctionParameters ")" |
        Id "(" ")"

FunctionName ::= <IDENTIFIER>

FunctionParameters ::= FunctionParameter (
        "," FunctionParameter )*

FunctionParameter ::= MathExpression |
        TableParameter

TableParameter ::= "[" TablePair
        ( "," TablePair )* "]"

TablePair ::= "(" SignedLiteral ","
        SignedLiteral ")"

Expression ::= ConditionalOrExpression

ConditionalOrExpression ::= ConditionalAndExpression (
```

```
        "||" ConditionalAndExpression )*

ConditionalAndExpression ::= EqualityExpression ( "&&"
        EqualityExpression )*

EqualityExpression ::= RelationalExpression (
        "==" RelationalExpression |
        "!=" RelationalExpression )*

RelationalExpression ::= UnaryExpression (
        "<" UnaryExpression |
        ">" UnaryExpression |
        "<=" UnaryExpression |
        ">=" UnaryExpression )*

UnaryExpression ::= MathExpression |
        "(" Expression ")"

Id ::= <IDENTIFIER>

Literal ::= ( <INTEGER_LITERAL> ) |
        ( <FLOATING_POINT_LITERAL> )

SignedLiteral ::= Literal |
        "-" Literal

IfStatement ::= <IF> "(" Expression ")"
        BlockStatement <ELSE> BlockStatement

BlockStatement ::= ( ReturnStatement |
        "{" ReturnStatement "}" )

ReturnStatement ::= ( MathExpression ";" |
        IfStatement )
```

And following is the token's definitions using `JavaCC` 's grammar definition language (tokens here is an expressions in `<>` braces)

```
TOKEN :
```

```
{
   < INTEGER_LITERAL: (["0"-"9"])+ >
|
  < FLOATING_POINT_LITERAL:
        (["0"-"9"])+ "." (["0"-"9"])*
      |
        "." (["0"-"9"])+
  >
}

TOKEN :
{
    < IF: "if" >
|
    < ELSE: "else" >
|
    < RETURN: "return" >
}

TOKEN : /* IDENTIFIERS */
{
  < IDENTIFIER: <LETTER> (<LETTER>|<DIGIT>)* >
|
  < #LETTER: [ "a"-"z", "A"-"Z", "_" ] >
|
  < #DIGIT: [ "0"-"9"] >
}
```