

JDecisiontable Manual

Table of Contents

Copyright of this document.....	2
Conventions	2
Overview.....	3
A first look.....	3
Left hand table.....	6
Right hand table.....	7
Short cuts.....	10
Cut, copy and paste.....	11
Menu Decisiontable.....	12
New, Open, Save and so on.....	12
New from Node Descriptions.....	13
Export to Csv.....	15
Menu Node.....	16
New Node.....	16
Remove Node.....	17
Copy Node.....	18
Menu Rule.....	21
New Rule.....	21
New Empty Rule.....	24
Remove Rule.....	25
Copy Rule.....	26
Menu Verify D.table.....	28
Run All Checks.....	28
Check Nodes_Dontcare.....	29
Check Rules_Disjunct.....	30
Check Rules_Number.....	31
Show actual num. of th. rules.....	32
Show expected num. of th. rules.....	32
Menu Tools.....	33
Clear Check Results.....	33
Sum Probabilities.....	33
Show Nodes w/o Y in Valid Rule.....	34
Menu Testspecification.....	36
Menu Options.....	37
Make Rows Higher.....	37
Use Icon Set 2.....	37
Use English.....	37
Menu Help.....	38
About.....	38

Help.....	38
Command Line.....	38
Option File.....	38
Linux.....	38
Windows®.....	38
File Formats.....	39
Hints for translators.....	41
Why there is no undo function and no autosave function too.....	44

Copyright of this document

Copyright 2012 - 2013 Michael Groß mgmechanics@mgmechanics.de

Copying and distribution of this file, with or without modification, are permitted in any medium without royalty provided the copyright notice and this notice are preserved. This file is offered as-is, without any warranty.

Conventions

Usually there are two screenshots for one action. The first screenshot shows the state before action and the later ones shows the result of the action.

First-class headlines starting with “Menu” represent the main menus, second-class headlines in this chapters the menu items.

“Disk” represents any volume build-in in your machine. In most machines this is your hard disk, in some machine it may a SD drive. It does not mean a removable volume i.e. an USB stick.

Overview

A first look

The image shows two screenshots of the JDecisiontable application. The top screenshot displays the 'rangeOfNumbers.5dt' decision table, and the bottom screenshot displays the 'dyeingSilk.5dt' decision table. Both tables have a 'cond?' column, a 'Description' column, a 'Probability' column, and a 'Comparison' column. The right side of each window shows a grid of values (Y, N, X) for various conditions.

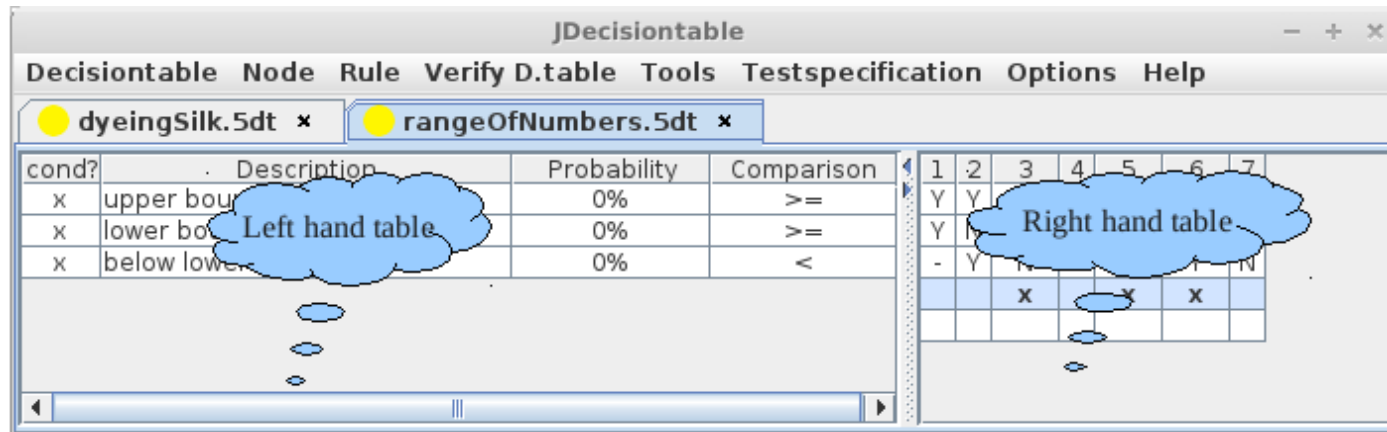
JDecisiontable - rangeOfNumbers.5dt

cond?	Description	Probability	Comparison
x	upper bound	0%	>=
x	lower bound	0%	>=
x	below lower bound	0%	<

JDecisiontable - dyeingSilk.5dt

cond?	Description	Probabi...	Com...
x	use blue color	0%	=
x	use red color	0%	=
	dyeing silk purple	0%	=
	dyeing silk blue	0%	=
	dyeing silk red	0%	=

This application can handle multiple tabs. Each tab contains one decision table.



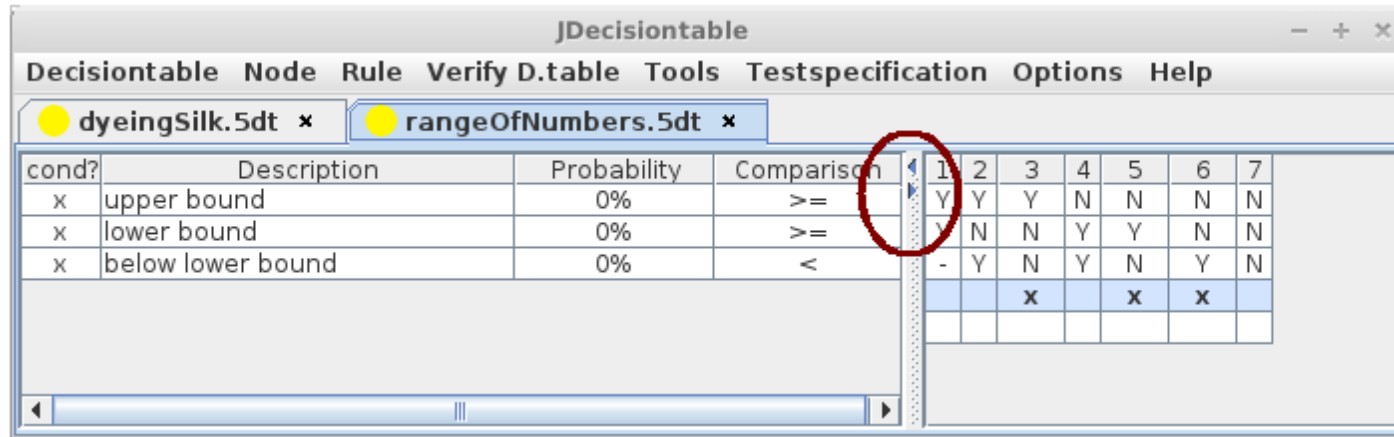
Each tab contains two tables.

- The left hand table contains the nodes
- The right hand table handles the rules

Left hand + right hand table = decision table.

Each decision table is independent from other decision tables and saved into one file.

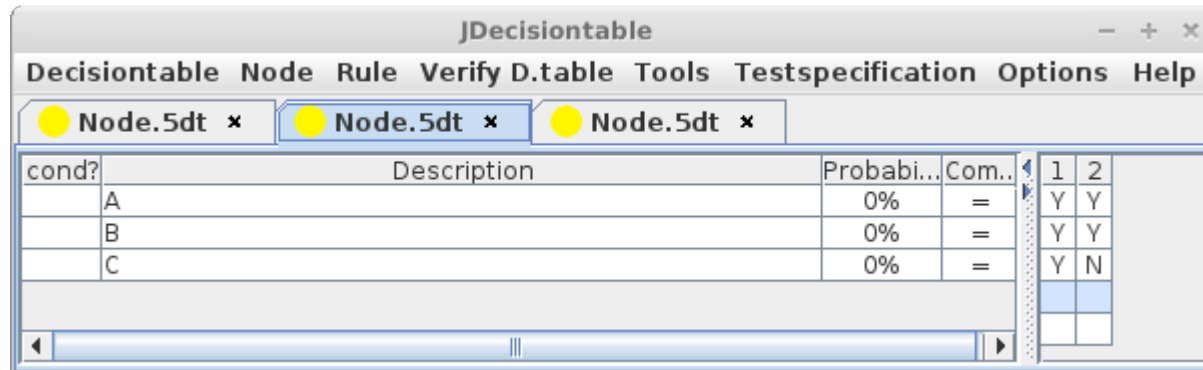
Both tables are divided by a bar which you can move to left or right. It comes with two arrows (red circle). Click on one of them to make the left or right hand table disappear.



You may also change the width of the columns in the left hand table. The width of columns in the right hand table is handled by the application.

Not all columns are editable as in a spreadsheet. These columns are limited to certain values. They change their value when you focus a cell and then type a certain key on your key board. This is pointed out in the next chapter.

You may open two or more decision tables with same file name at once:



Left hand table

The left hand table handles only the nodes of the decision table. Nodes are organized in rows. Each row shows the parts (fields) of one node.

Field	Purpose	Editable?	Limited?	Limited to
cond?	[x] = this node is a condition [] = this node is an action “cond?” means “is condition”	No	Yes	“x”, DEL key, BACKSPACE key [←]
Description	Contains either what the condition is or which action should performed.	Yes	No	I would avoid to include tab [⇧] and linefeed chars here. There is really no need to format text here. You may use the option to make the rows higher to distinct nodes visibly.
Probability	Contains the probability that this condition is true or that this action is performed.	Yes	Yes	Numbers, decimal delimiter and “%” char i.e. “2” → 2.0%, “2.1” → 2.1%, “3%” → 3.0 % If you leave the field empty it will be filled with default value again. Same if you type some chars outside the limits (i.e. a letter) and leave the field.
Comparison	Contains the comparison. It is mainly used when dealing with ranges - see screenshots .	No	Yes	Type a number from range 1..5. 1 → “>=”, 2 → “<=”, 3 → “=”, 4 → “>”, 5 --> “<” ¹

1 Do you see the pattern? ‘=’ is in the middle while ‘>=’ and ‘<=’ are left from it and ‘>’ and ‘<’ stay to the right.

There is an easy trick to delete existing values in editable fields:

- select the cell
- type Backspace key [←] several times until the selected cell is empty
- type new values

Right hand table

The right hand table handles the rules of the decision table. Rules are organized in columns. Each column contains the parts (fields) of one rule.

The table header contains the number of each rule. If you insert or remove a rule they get new numbers.

Below the numbers there are **decisions** (hatched area). Depending on internationalization each letter or char means:

Letter / char	Meaning / decision
Y	Yes
N	No
-	Don't Care

The screenshot shows the JDecisiontable application window. It has a menu bar with 'Decisiontable', 'Node', 'Rule', 'Verify D.table', 'Tools', 'Testspecification', 'Options', and 'Help'. Below the menu bar are two tabs: 'dyeingSilk.5dt' and 'rangeOfNumbers.5dt'. The main area is divided into two parts. On the left is a table with columns 'cond?', 'Description', 'Probabi...', and 'Com...'. It contains five rows of data related to dyeing silk. On the right is a large table with 13 columns numbered 1 to 13. The first row of this table contains 'Y' in columns 1-4, 'M' in 5-7, 'N' in 8, 'K' in 9-12, and 'M' in 13. The second row contains 'Y' in 1, 'V' in 2, 'N' in 3, 'N' in 4, 'N' in 5, 'Y' in 6, 'Y' in 7, 'Y' in 8, 'Y' in 9, 'M' in 10, 'M' in 11, 'M' in 12, and 'M' in 13. The third row contains 'Y' in 1, 'N' in 2, 'Y' in 3, 'N' in 4, 'N' in 5, 'Y' in 6, 'M' in 7, 'M' in 8, 'M' in 9, 'M' in 10, 'M' in 11, 'M' in 12, and 'M' in 13. The fourth row contains 'Y' in 1, 'N' in 2, 'Y' in 3, 'N' in 4, 'N' in 5, 'Y' in 6, 'M' in 7, 'M' in 8, 'M' in 9, 'M' in 10, 'M' in 11, 'M' in 12, and 'M' in 13. The fifth row contains 'Y' in 1, 'N' in 2, 'Y' in 3, 'N' in 4, 'N' in 5, 'Y' in 6, 'M' in 7, 'M' in 8, 'M' in 9, 'M' in 10, 'M' in 11, 'M' in 12, and 'M' in 13. The sixth row contains 'Y' in 1, 'N' in 2, 'Y' in 3, 'N' in 4, 'N' in 5, 'Y' in 6, 'M' in 7, 'M' in 8, 'M' in 9, 'M' in 10, 'M' in 11, 'M' in 12, and 'M' in 13. The seventh row contains 'Y' in 1, 'N' in 2, 'Y' in 3, 'N' in 4, 'N' in 5, 'Y' in 6, 'M' in 7, 'M' in 8, 'M' in 9, 'M' in 10, 'M' in 11, 'M' in 12, and 'M' in 13. The eighth row contains 'Y' in 1, 'N' in 2, 'Y' in 3, 'N' in 4, 'N' in 5, 'Y' in 6, 'M' in 7, 'M' in 8, 'M' in 9, 'M' in 10, 'M' in 11, 'M' in 12, and 'M' in 13. The ninth row contains 'Y' in 1, 'N' in 2, 'Y' in 3, 'N' in 4, 'N' in 5, 'Y' in 6, 'M' in 7, 'M' in 8, 'M' in 9, 'M' in 10, 'M' in 11, 'M' in 12, and 'M' in 13. The tenth row contains 'Y' in 1, 'N' in 2, 'Y' in 3, 'N' in 4, 'N' in 5, 'Y' in 6, 'M' in 7, 'M' in 8, 'M' in 9, 'M' in 10, 'M' in 11, 'M' in 12, and 'M' in 13. The eleventh row contains 'Y' in 1, 'N' in 2, 'Y' in 3, 'N' in 4, 'N' in 5, 'Y' in 6, 'M' in 7, 'M' in 8, 'M' in 9, 'M' in 10, 'M' in 11, 'M' in 12, and 'M' in 13. The twelfth row contains 'Y' in 1, 'N' in 2, 'Y' in 3, 'N' in 4, 'N' in 5, 'Y' in 6, 'M' in 7, 'M' in 8, 'M' in 9, 'M' in 10, 'M' in 11, 'M' in 12, and 'M' in 13. The thirteenth row contains 'Y' in 1, 'N' in 2, 'Y' in 3, 'N' in 4, 'N' in 5, 'Y' in 6, 'M' in 7, 'M' in 8, 'M' in 9, 'M' in 10, 'M' in 11, 'M' in 12, and 'M' in 13. The bottom row of the table contains 'x' in columns 1, 4, 8, and 13. Below the table is a label 'whiteSilk.5dt'. A yellow callout bubble labeled 'Rule number' points to the first column of the table. Another yellow callout bubble labeled 'Decisions' points to the hatched area of the table.

Some fields are not editable but change their value when you type certain keys on your keyboard.

Field	Purpose	Editable ?	Limited?	Limited to
decision	Contains the decisions of each rule.	No	Yes	"y", "n", "-"
isValid	[x] = this rule is valid [] = this rule is not valid	No	Yes	"x", DEL key, BACKSPACE key [←]
Successor	Contains the successor. This another decision table to continue after finishing with this rule.	Yes	No	

Short cuts

All menu items used for day-by-day work may also be used by using a short cut.

“Ctrl-Q” means “press down the Control key, type “w” and release Control key.

“Ctrl+Alt-W” means “press down the Control key, press down Alt key, type “w”, release Alt key and release Control key.

“Ctrl+Shift-W” means “press down the Control key, press down Shift key, type “s”, release Shift key and release Control key.

Please note that “Ctrl-Q” does not mean to type an upper case Q – just type the key labeled with “Q”.

You may

- switch between left hand and right hand table using Ctrl-# (Mnemonic: # = table)
- navigate from cell to cell using Arrow keys, Tab, Shift-Tab, Enter and Shift-Enter
- start editing editable cells by going to this cell and just start typing, typing F2 or double-click

Currently there is no short cut to go from tab to tab. It would be easy to implement but it seems not possible to avoid any short cut used to switch from desktop to desktop.

Some short cuts are different than in well-known Windows applications i.e. Ctrl-Q (q stands for quit) rather than Alt-F4. The author was free to keep along common Linux desktop applications.

Cut, copy and paste

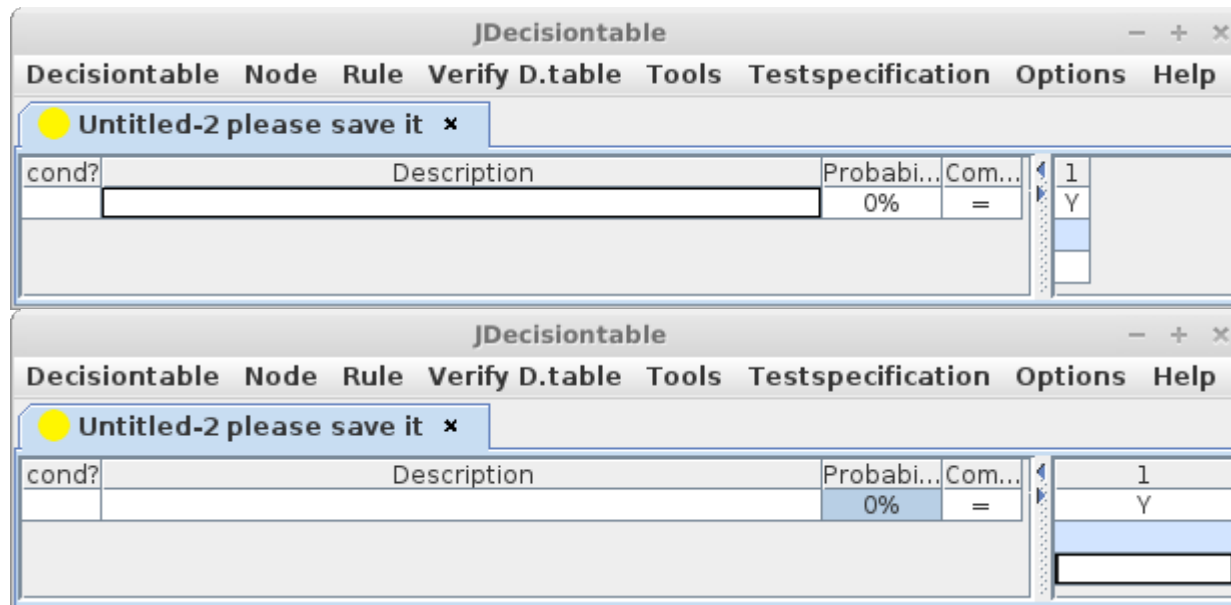
Cut, copy and paste works fine with JDecisiontable in Linux and Windows® (both tested with JDecisiontable version 1.1.1) for the fields

Description and **Successor** (only) if you bring these fields to edit mode i.e. by

- double click
- type F2
- type any letter, number, ... and place the cursor into the cell by clicking somewhere in the cell

Then use Ctrl+X to cut, Ctrl+C to copy and Ctrl+V to paste some text from/into the selected cell as usual.

The screenshots below show JDecisiontable with fields Description and Successor being in the edit mode.



Menu Decisiontable

New, Open, Save and so on

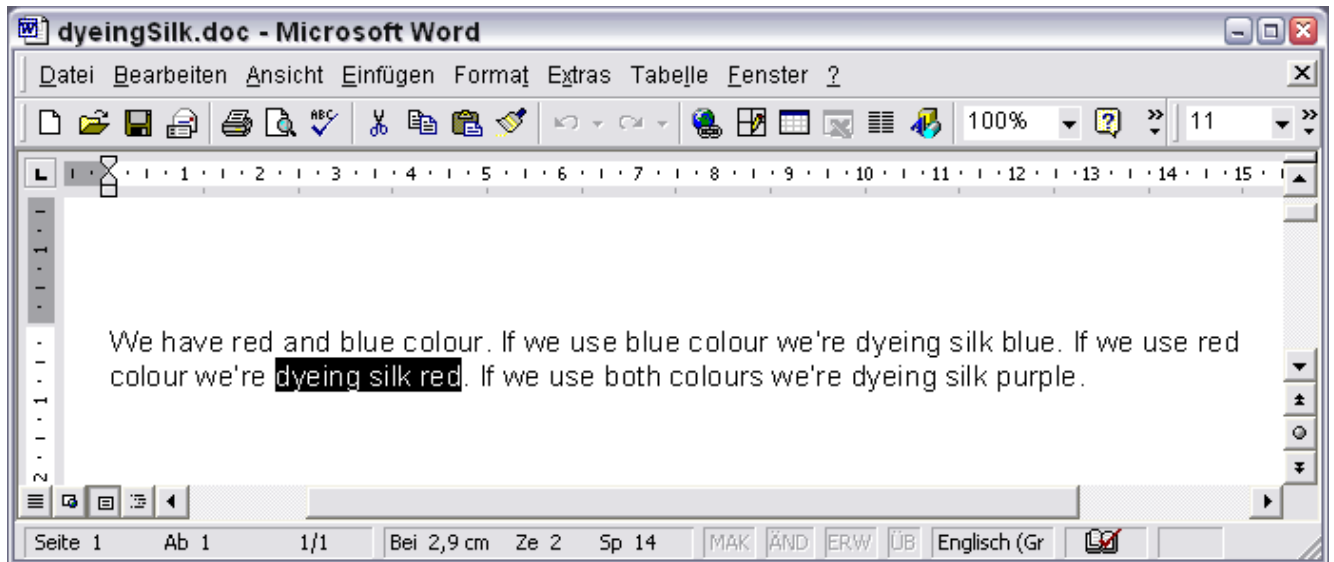
This menus should work as you expect it from other applications with one exception: There is intentionally no decision table open when JDecisiontable just started. Please use Open or New to get one.

Menu item	Function
New	Creates a new decision table.
Open	Opens a dialog to open a decision table. The file name suffix for decision tables is *.5dt . This are the only kind of files which you can open with JDecisiontable! There are no facilities in JDecisiontable to open files as *.5ts (test specifications) or *.csv! These are for processing with other applications.
Close	Closes the decision table in the tab which is active yet. If you've changed the decision table but have not saved the changes yet it will ask you to save the changes now or to close the table without saving. The short cut is same as in Firefox (for Linux?) and Notepad++.
Close All	Closes all decision table in all tabs. If one table was changed but not saved yet it asks you to save your changes.
Save	If the decision table was created but not saved to disk it will raise a dialog where you may save the decision table. Otherwise it will save it quietly.
Save As	It will always raise a dialog where you may save the decision table. You do not need to type the file name suffix (*.5dt) after the file name – this is done automatically if the suffix is missed.
Save All	It will save all decision tables to disk. If one decision table was created but not saved to disk it will raise a dialog where you may save the decision table.
Exit	Closes all decision tables and quits the application. If one decision table was changed but not saved to disk yet it will ask you if you want to save the decision table.

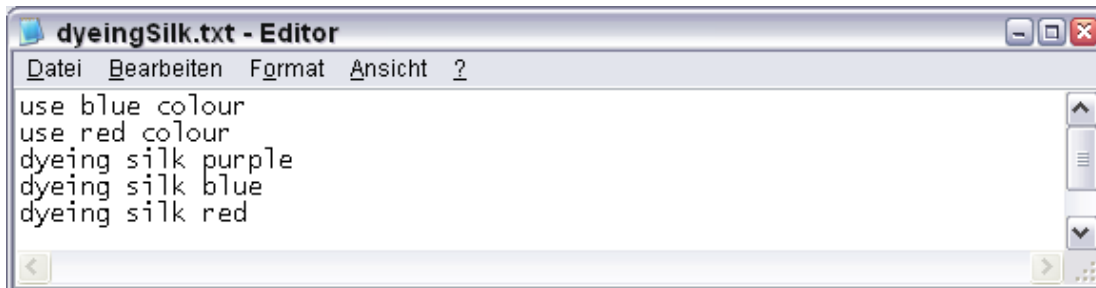
New from Node Descriptions

This menu item let you create a new decision table using a text file to read the node descriptions from. One line is used as one description so here will be as much nodes as there are lines in the text file. All fields except Decision will have same default values as with Decisiontable → New. Also there is one rule only starting with “Yes”. How can you use it?

²First, copy some text from word processor or spread sheet i.e. Microsoft® Word® 2000:

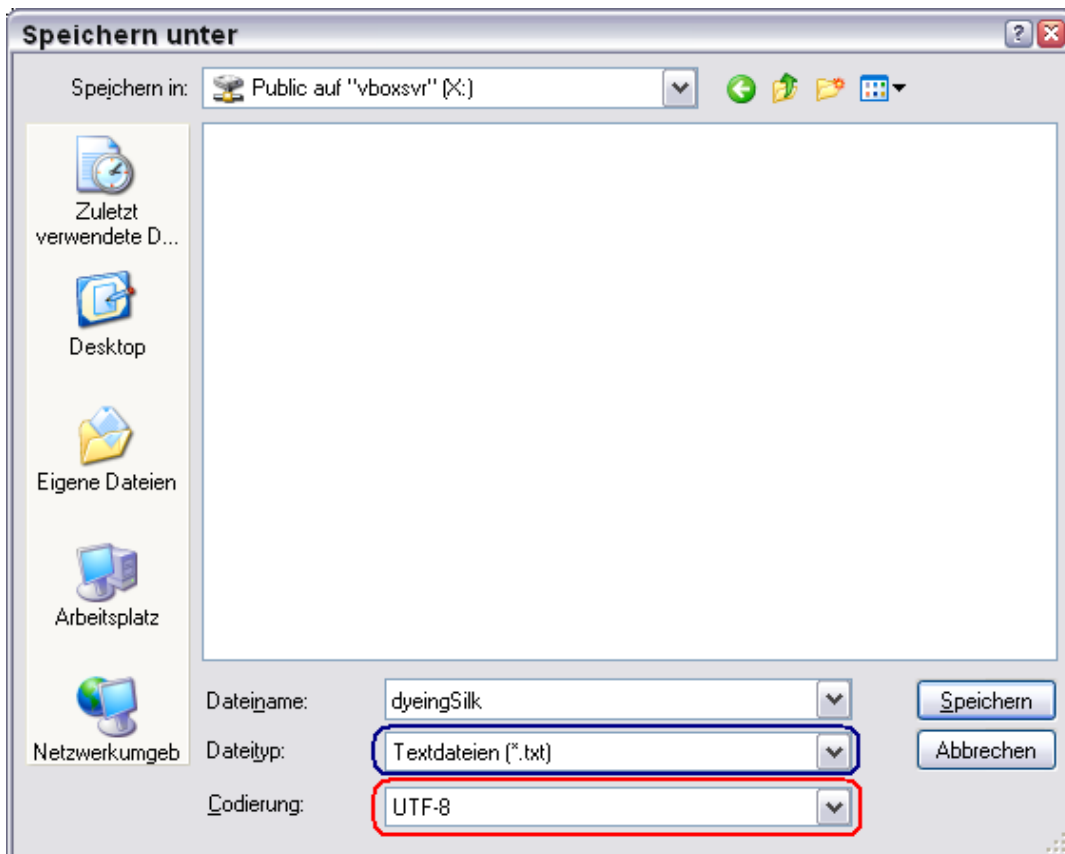


Then paste it to a text editor i.e. Accessories → Editor in Microsoft® Windows®:

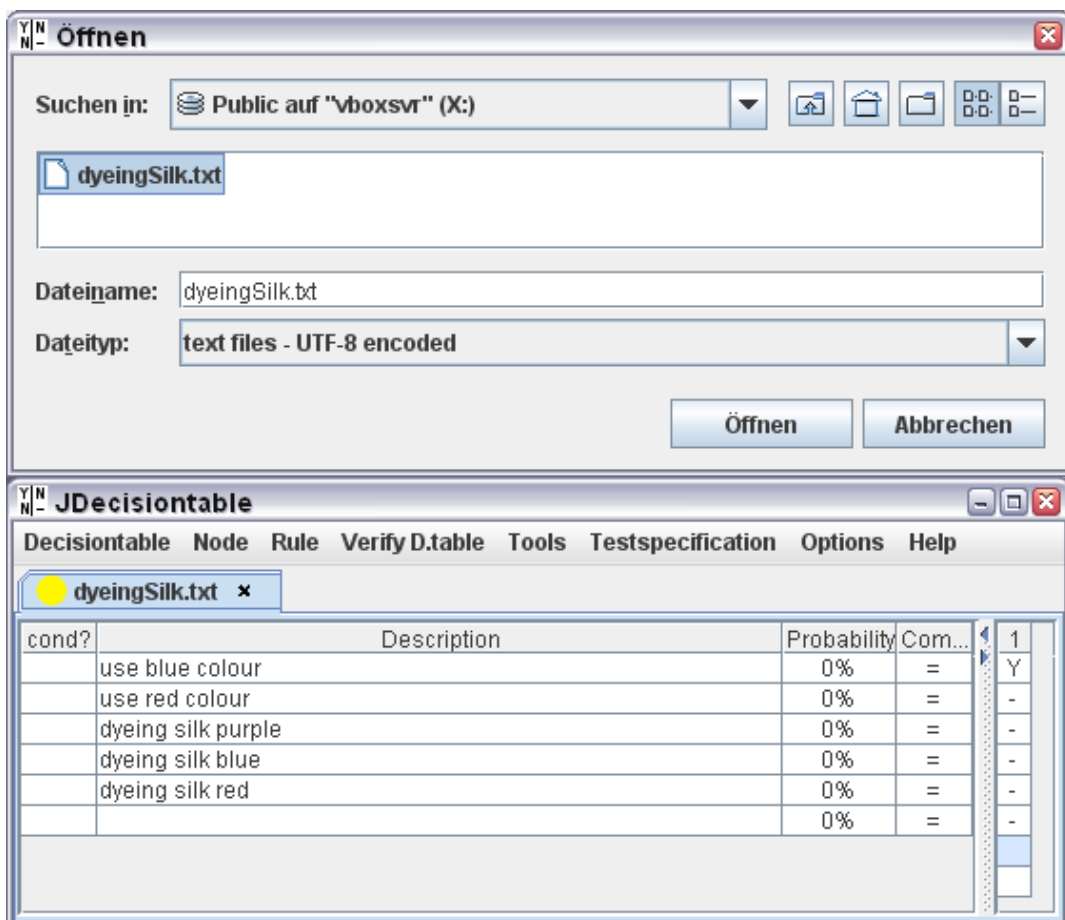


Now save the file using format “Text” and encoding “UTF-8” (this file is called “the text file”). You do not need to type the file name suffix:

2 This way may doesn't look straight-forward but is (almost) fail-safe. Don't use “ANSI” as encoding. See “File Formats” for details.



At last go to JDecisiontable and open the text file with Decisiontable → New from Node Descriptions:



Please do not forget to save the file as decision table now! The suffix "txt" in the tab is left with intend to remember this step.

Export to Csv

This will write the decision table in the current active tab to a csv file. A File Save dialog will appear so you may give it a path there to save the file. Please note that there is no facility in this application to read it in again. Its for other application only. There is no way back³!

Before generating the file it will run all checks – same as you choose Verify D.table → Run All Checks. If the check passed it will just write “VALID” in cell 1,1. Otherwise there will written “NOT VALID” in this cell. The application will not inform you in this case (no message pops up ...).

And this is what the files looks like (left example generated from rangeOfNumbers.5dt).

VALID				1	2	3	4	5	6	7
x	upper bound	0.0	>=	Y	Y	Y	N	N	N	N
x	lower bound	0.0	>=	Y	N	N	Y	Y	N	N
x	below lower bound	0.0	<	-	Y	N	Y	N	Y	N
					x		x	x		

NOT VALID				1	2	3	4	5	6	7
x	upper bound	0.0	>=	Y	Y	Y	N	N	N	N
x	lower bound	0.0	>=	Y	N	N	Y	Y	N	N
x	below lower bound	0.0	<	Y	Y	N	Y	N	Y	N
					x		x	x		

Please see chapter “File Formats” for a technical specification of this files.

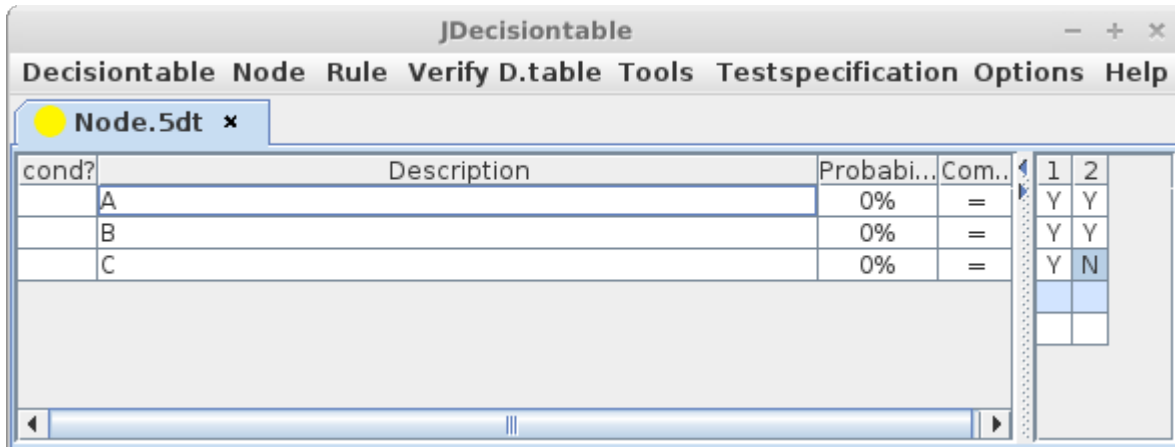
³ Since the regular file format is JSON you may write your own class or script which reads this csv into proper JSON. See “File Formats”.

Menu Node

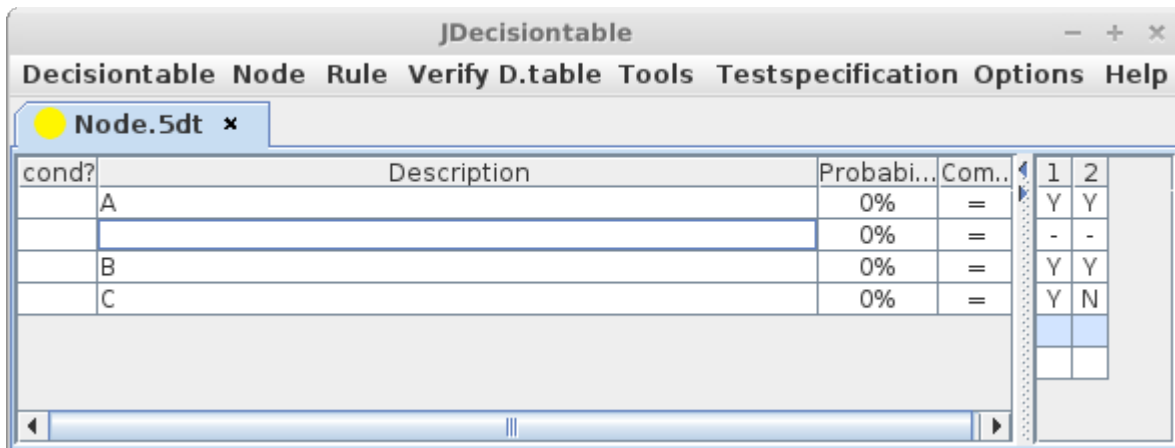
New Node

This creates a new node below the row of the cell which is selected.

Please make sure that there is a cell selected in left hand table as shown below to make sure that the new node is inserted on the expected place.⁴

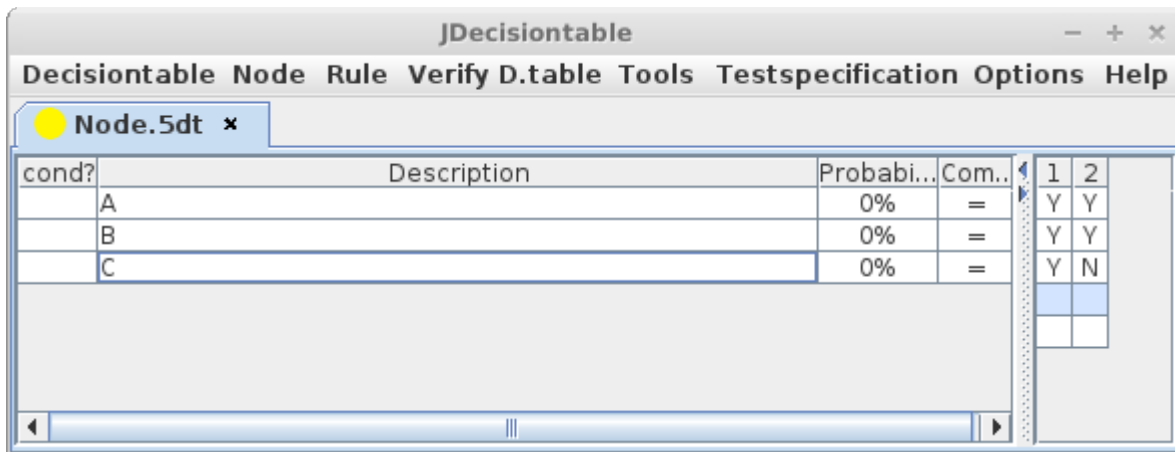


Of course this creates also a new row in all rules. This row is filled with Don't Care decisions.

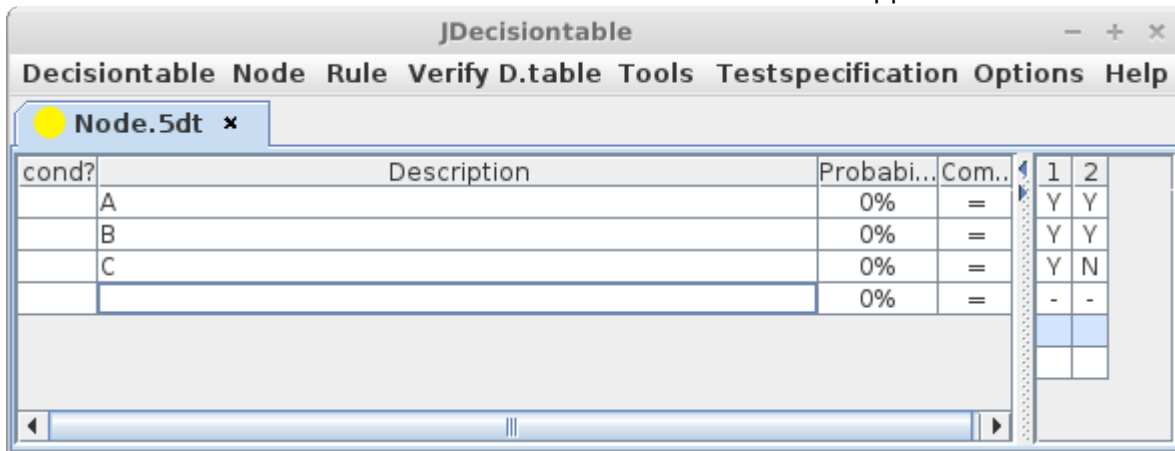


⁴ If you click elsewhere in the application the left table will loose the focus. But it will still remember the cell selected at last. One could say: When left table lost the focus take selected cell from right table (assuming that the right hand table will have the focus). But if you click on a menu both tables loose the focus and both remember the last selected cell. Which cell is the one you want a new node below?

As a fall back if there was no cell selected since table was opened it will insert the new node below the last node.



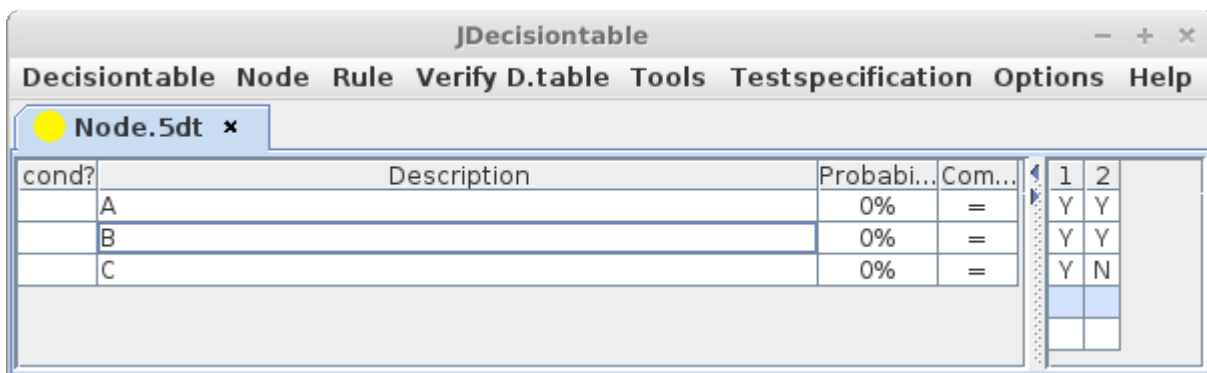
This works of course also for the last decision. The new node is append after the last row.



Remove Node

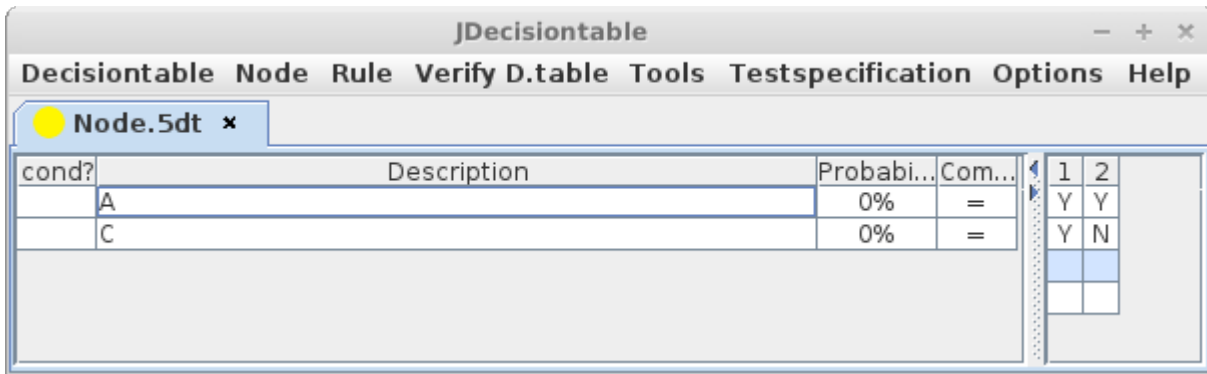
Removing nodes works same way as create new ones. Just select a node and select Node → Remove Node to remove it.

Please make sure that there is a cell selected in left hand table as shown below to make sure that the expected node is removed.⁵

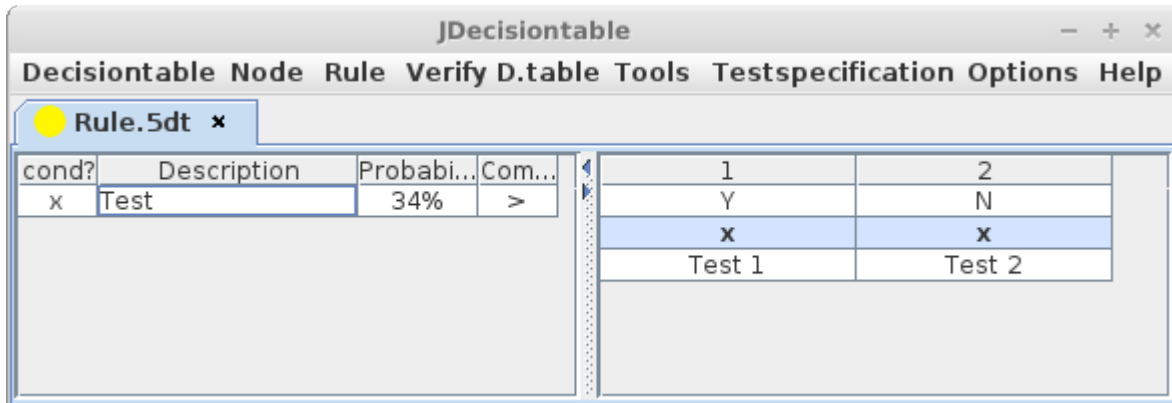


And of course it will remove the cells in same row in right hand table.

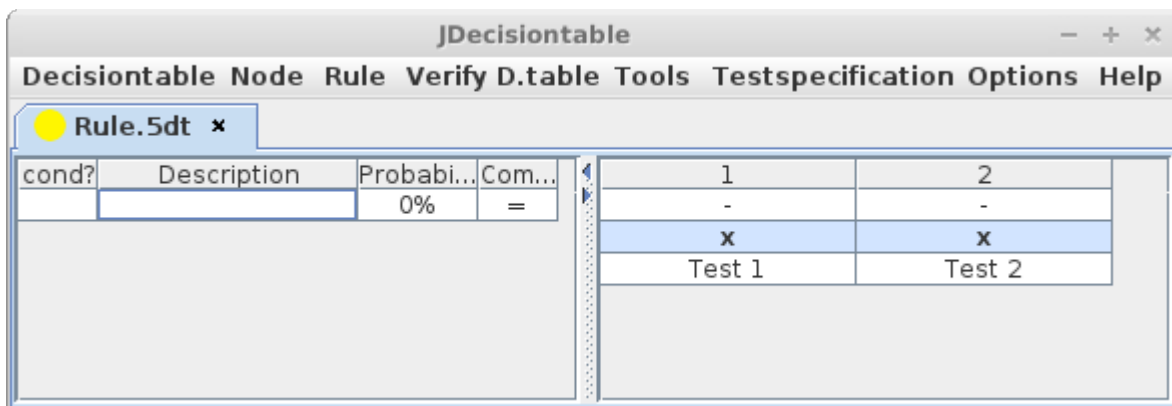
⁵ For same reasons as in Node → New Node. And as in Node → New node there is same fall back: If there was no cell selected since table was opened it will remove the last node.



Q: What would happen if I remove the last remaining node?



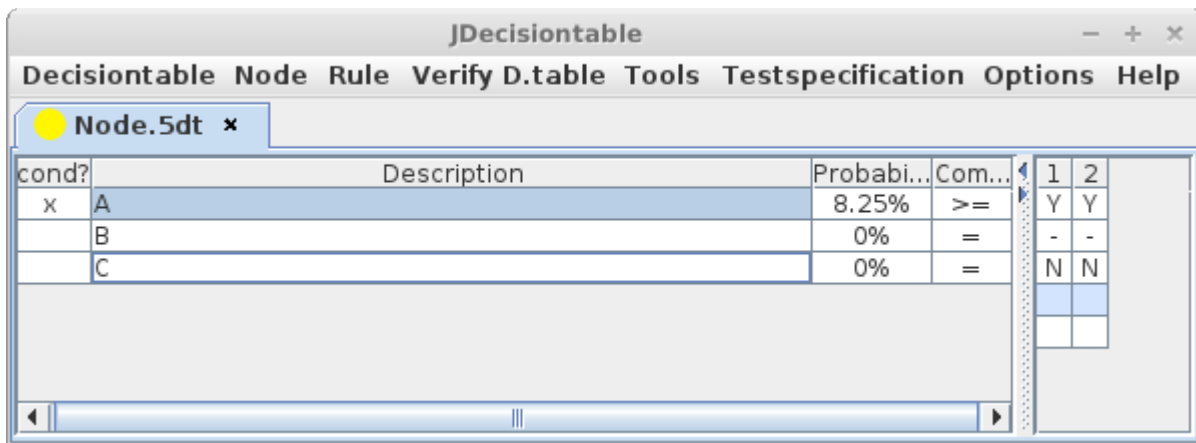
A: JDecisiontable accepts your wish and removes this node but it creates a new one immediately. The rules remain but the decisions will be turned to Don't Care-decisions.



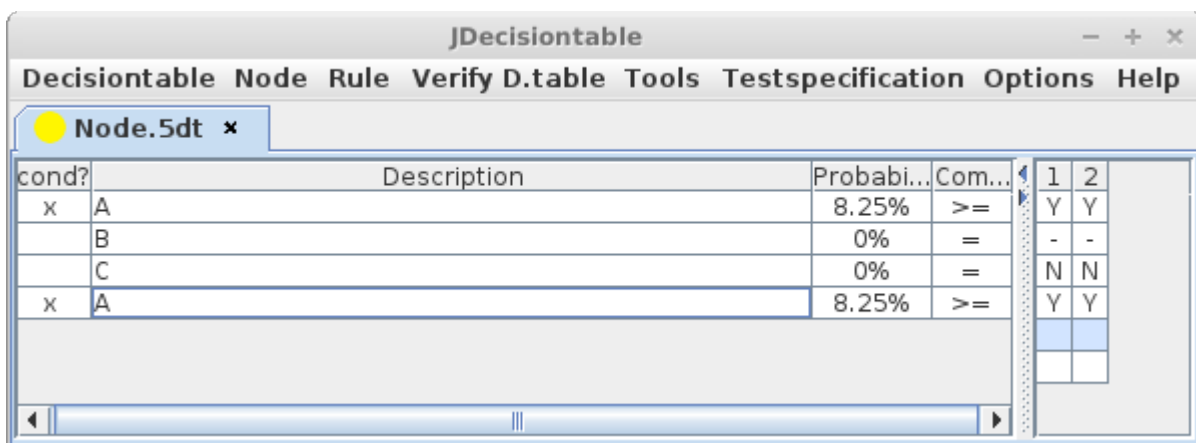
Copy Node

To copy a file you need a source and a destination, isn't it? Same as for a node: You need a node to duplicate (= source) and a place where to insert the copy. You can tell these two items by selecting two nodes. The upper one will be recognized as source and the lower one as destination⁶. Like this:

⁶ To be the destination means that the copy will be inserted below the selected cell.



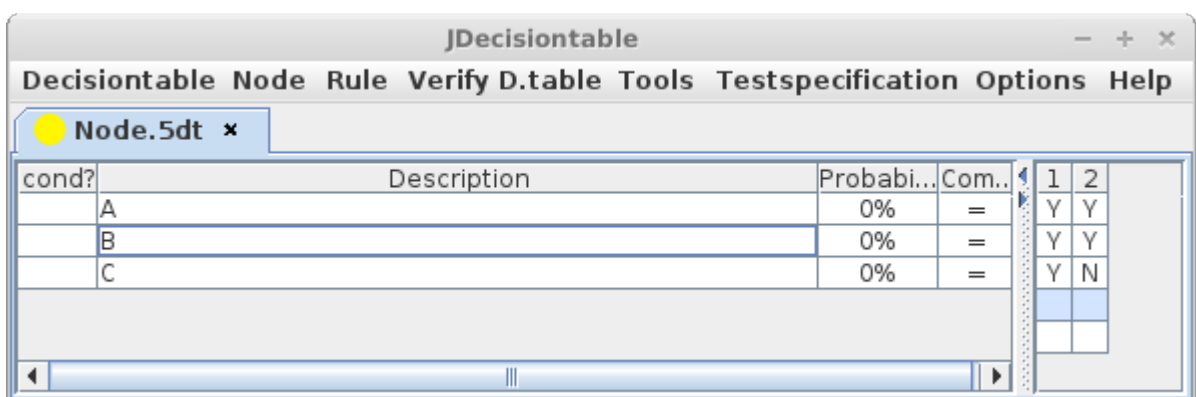
You may select more than one cell by hold down Ctrl key while selecting. Please make sure to select cells in left hand table (only) to get the expected result⁷.



Of course are the decisions of the selected node copied with this node.

The column of the selected cells doesn't matter - selecting the cell in the first row in column Condition and the cell in second row in column Comparison gives same result as selecting the cells in first and second row in column Description. It also doesn't matter which cell was selected first (in time). Only the row matters - if the cell is in left hand table.

You may also select a single cell:



⁷ For same reason as Node → New Node. Fall back if no cell was selected before: It will copy the last node and append it below the last node.

In this case the selected cell is source and destination so a copy of this node will be inserted below this node.

The screenshot shows the JDecisiontable application window with the 'Node.5dt' tab selected. The main table has columns: 'cond?', 'Description', 'Probabi...', and 'Com...'. The row with 'B' in the 'Description' column is selected. To the right of the main table is a smaller table with columns '1' and '2'.

cond?	Description	Probabi...	Com...
	A	0%	=
	B	0%	=
	B	0%	=
	C	0%	=

1	2
Y	Y
Y	Y
Y	Y
Y	N

Selecting a range of cells results in copying the most upper selected cell below the last selected cell⁸:

The screenshot shows the JDecisiontable application window with the 'Node.5dt' tab selected. The main table has columns: 'cond?', 'Description', 'Probabi...', and 'Com...'. The rows with 'A', 'B', and 'C' in the 'Description' column are selected. To the right of the main table is a smaller table with columns '1' and '2'.

cond?	Description	Probabi...	Com...
	A	0%	=
	B	0%	=
	C	0%	=
	D	0%	=

1	2
Y	Y
Y	Y
Y	N
-	-

By the way - the focus is always set automatically to the copy:

The screenshot shows the JDecisiontable application window with the 'Node.5dt' tab selected. The main table has columns: 'cond?', 'Description', 'Probabi...', and 'Comp...'. The rows with 'A', 'B', 'C', 'A', and 'D' in the 'Description' column are visible. The row with 'A' (the second one) is selected. To the right of the main table is a smaller table with columns '1' and '2'.

cond?	Description	Probabi...	Comp...
	A	0%	=
	B	0%	=
	C	0%	=
	A	0%	=
	D	0%	=

1	2
Y	Y
Y	Y
Y	N
Y	Y
-	-

⁸ "upper" and "last" means the row of the selected cells.

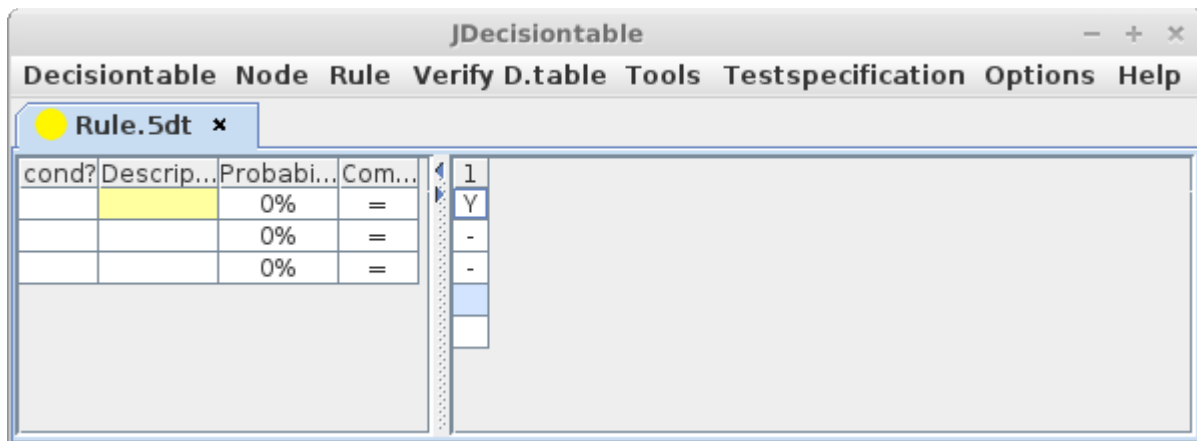
Menu Rule

While there are two ways to get a new node (New Node and Copy Node) there are three ways to get a new rule:

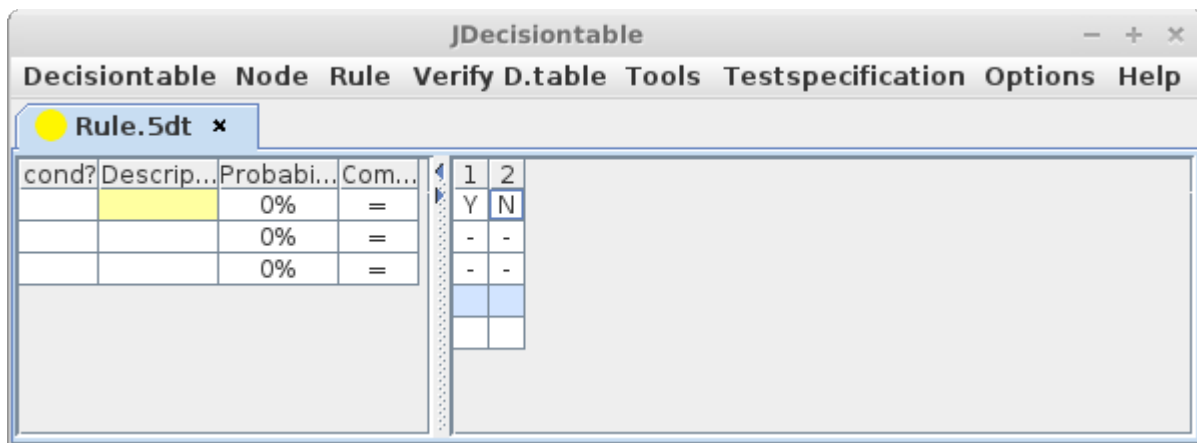
- **New Empty Rule** is similar to New Node
- **Copy Rule** is similar to Copy Node
- **New Rule** and New Node are “false friends” since they do completely different things⁹

New Rule

New Rule does not create a rule - it copies a rule and modifies the copy! Like this:

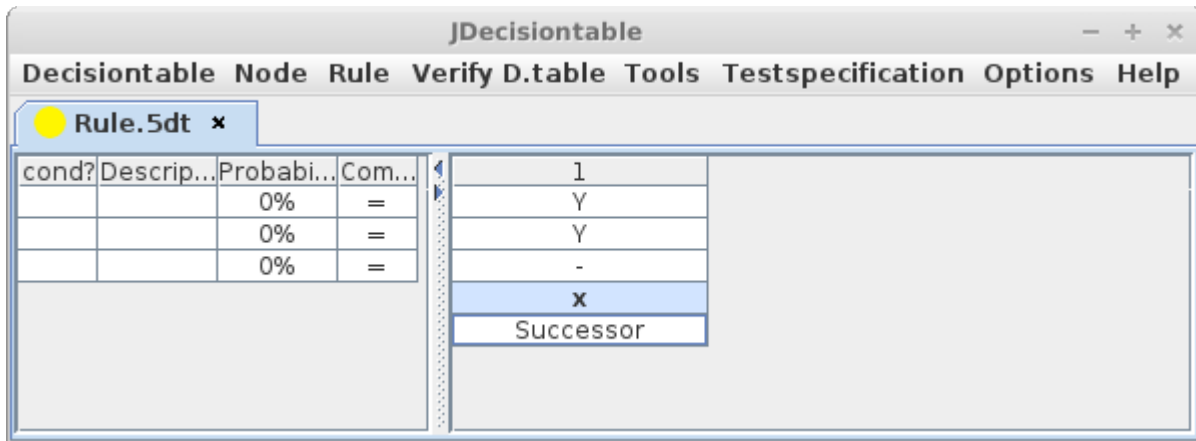


Rule 1 is the source and 2 is the copy.

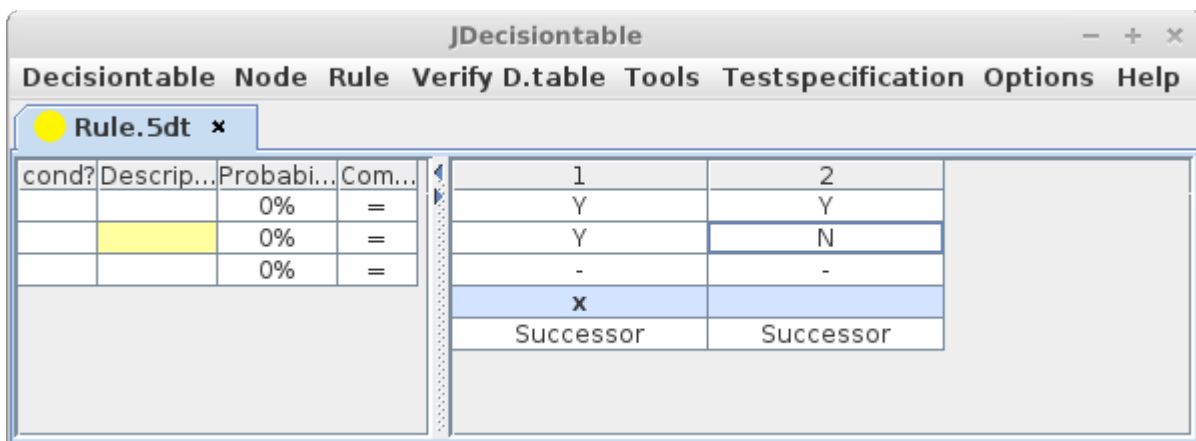


You don't believe that 2 is a modified copy of 1? Ok, lets do again:

⁹ The name “New Rule” is still given with intend because you will use this menu to get a new rule all the time. Probably you will forget what New Empty Rule does because you never use it.

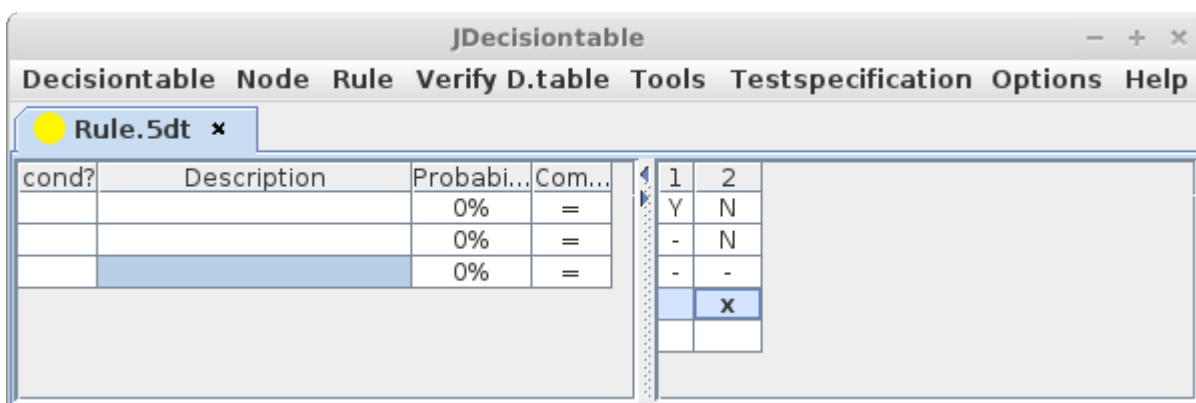


It's really a copy:



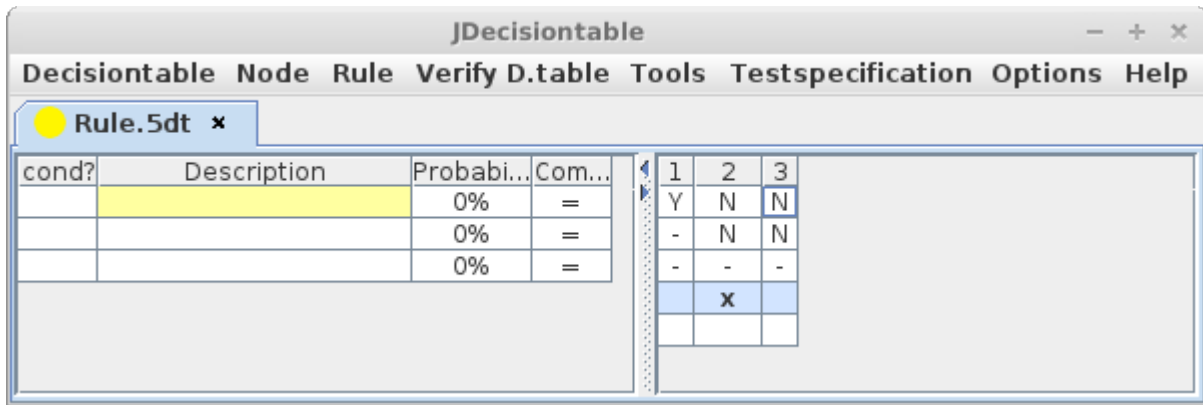
This is the most important feature for creating this kind of decision tables for which JDecisiontable was made for. It copies the selected rule and changes the last Yes-decision into a No-decision. The new rule is always assumed to be invalid. After completing this rule you may mark it as valid.

If there is no Yes-decision in the selected rule it does the same as Rule → Copy Rule¹⁰. It copies the rule and does not change anything but the isValid mark.



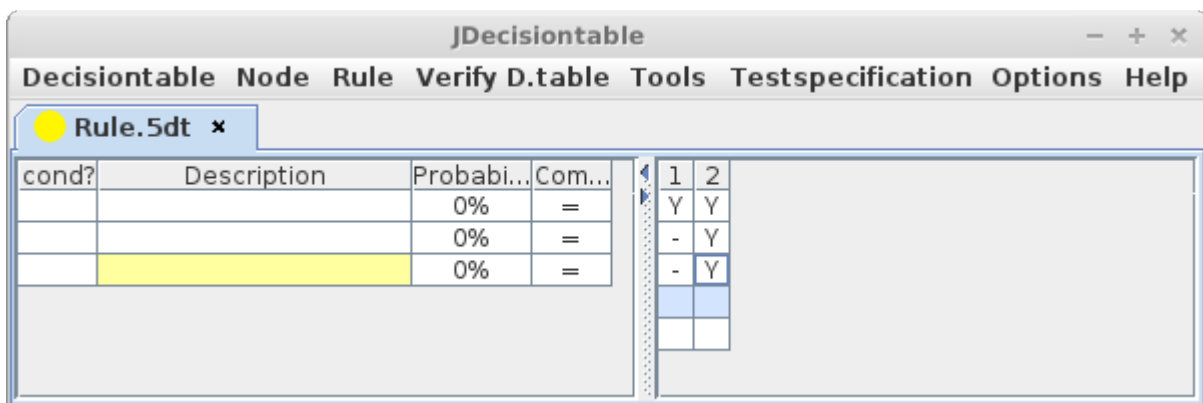
Rule 3 is the modified copy of node 2. If a rule is valid the column is automatically wider than the column for invalid rules.

¹⁰ It makes no sense to create a new rule from a rule without Yes-decision since your decision table is complete if you got a rule by Rule → New Rule with No- and Don't Care-decisions only. (Assumed that you did follow the method as described in "How to make decision tables" and did no mistakes.)

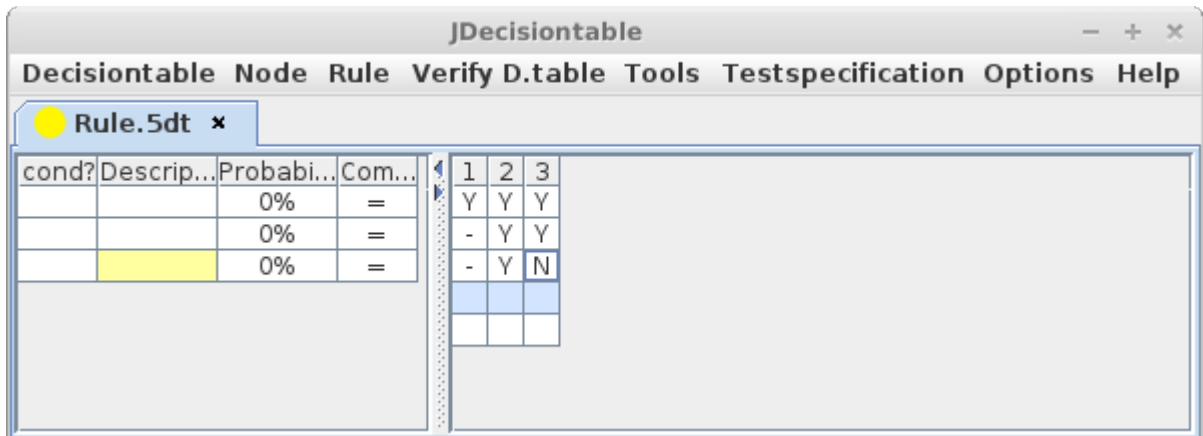


To improve the workflow the changed decision in the new rule will be selected automatically.

The source is always the selected rule:

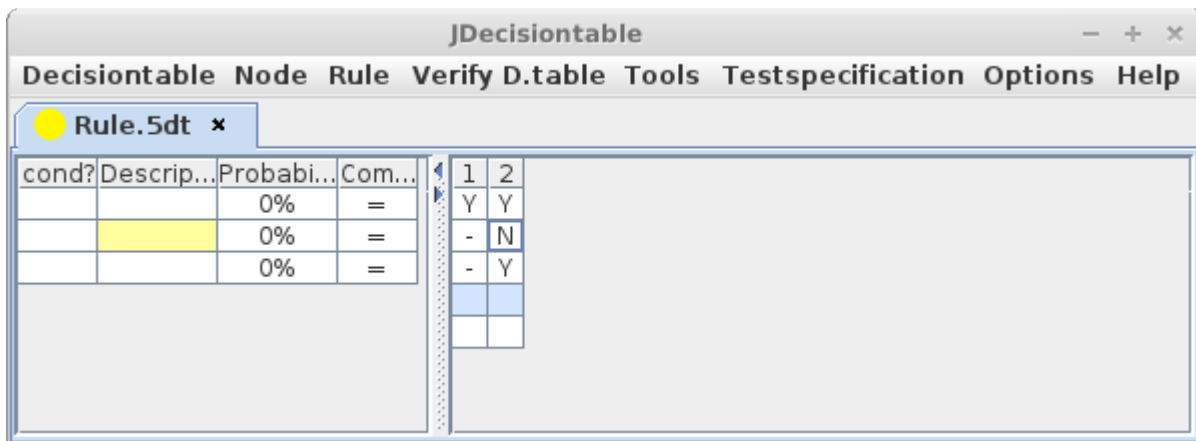


Now rule 2 was the source and rule 3 is the modified copy. As always, the last Yes decision in the copy was changed into a No decision.

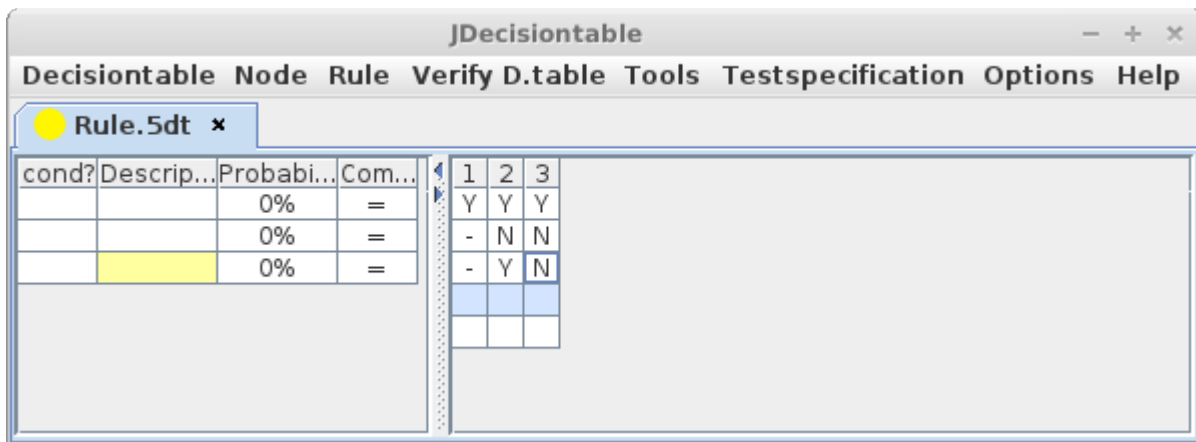


You may of course select any rule. The new rule will be inserted right from the selected rule.

Q: What happens if there is already a No decision in the rule?



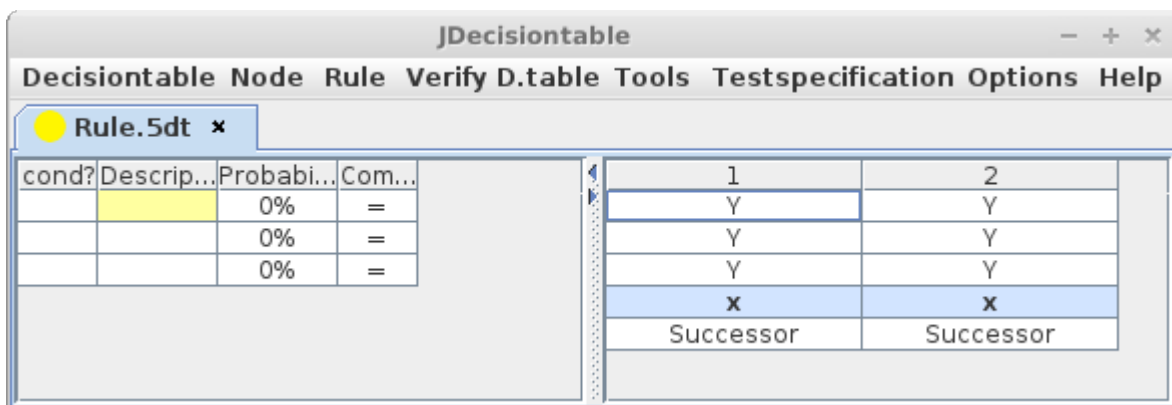
A: It still changes the last Yes decision in the copy.



Using this menu is the standard way to get new rules. All other menus are for correcting mistakes or inserting a new node after the decision table was done. But the latter is error prone. The author got better results by copying the old decision table, removing all rules and creating new rules while having a look on the valid rules of the old decision table.

New Empty Rule

This menu inserts a new created rule right from the selected rule. Other as in Rule → New Rule the selected rule is not copied.



The default value for decisions is Don't Care.

JDecisiontable						
Decisiontable Node Rule Verify D.table Tools Testspecification Options Help						
● Rule.5dt ✕						
cond?	Descrip...	Probabi...	Com...	1	2	3
		0%	=	Y	-	Y
		0%	=	Y	-	Y
		0%	=	Y	-	Y
				x		x
				Successor		Successor

Remove Rule

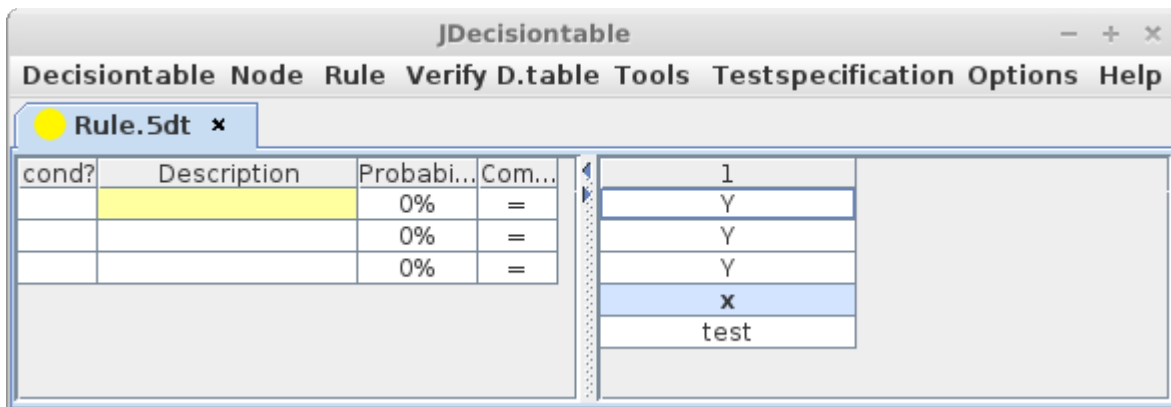
Remove Rule remove the selected rule.

JDecisiontable						
Decisiontable Node Rule Verify D.table Tools Testspecification Options Help						
● Rule.5dt ✕						
cond?	Description	Probabi...	Com...	1	2	
		0%	=	Y	N	
		0%	=	-	-	
		0%	=	-	-	

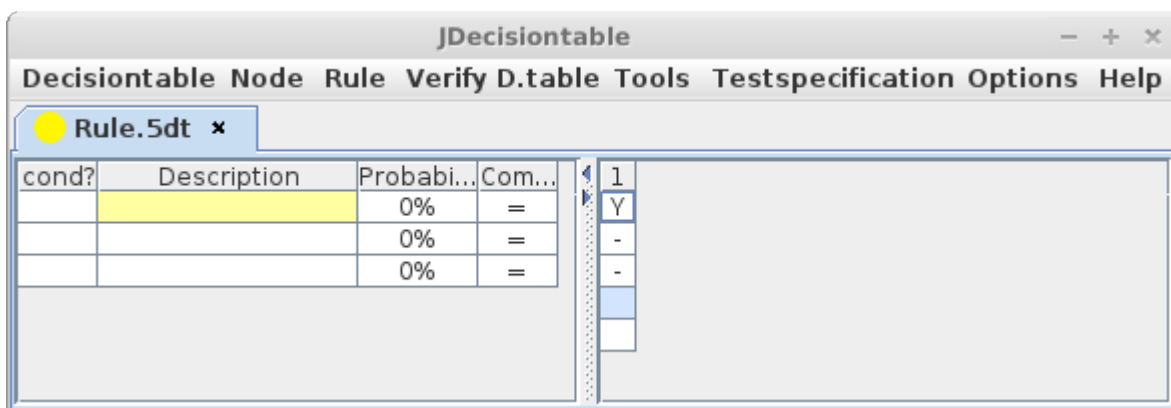
Removed!

JDecisiontable						
Decisiontable Node Rule Verify D.table Tools Testspecification Options Help						
● Rule.5dt ✕						
cond?	Description	Probabi...	Com...	1		
		0%	=	N		
		0%	=	-		
		0%	=	-		

Q: What if I remove the last remaining rule from a decision table?

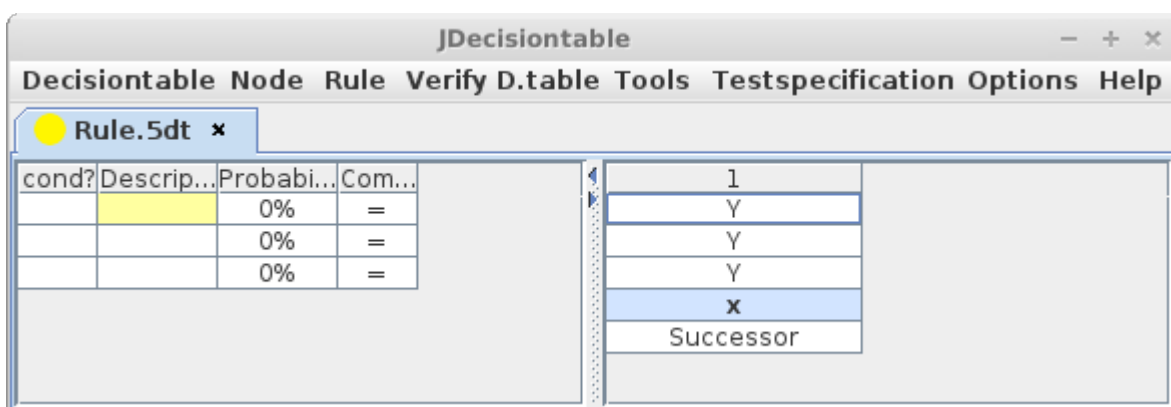


A: JDecsiontable will remove this rule but create a new empty rule soon. The it sets a Yes-decision in the first cell¹¹.



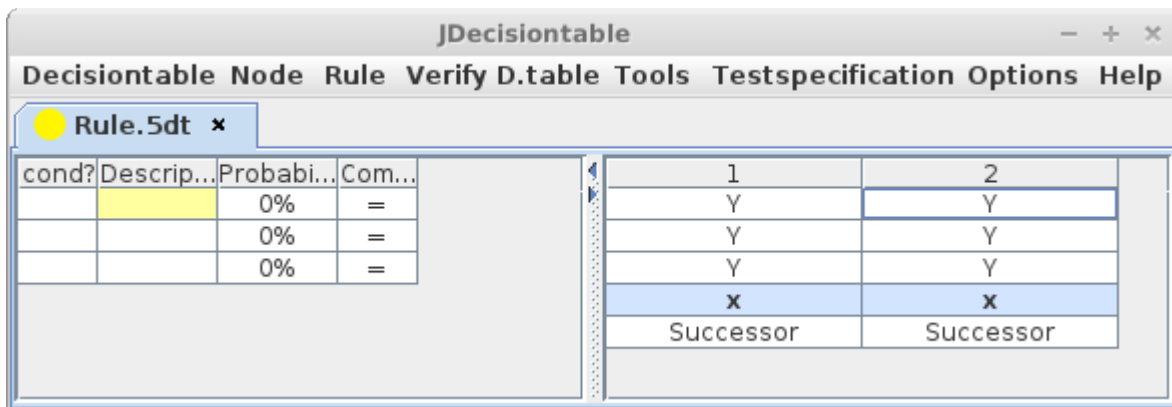
Copy Rule

Copy Rule just copies the selected rule.

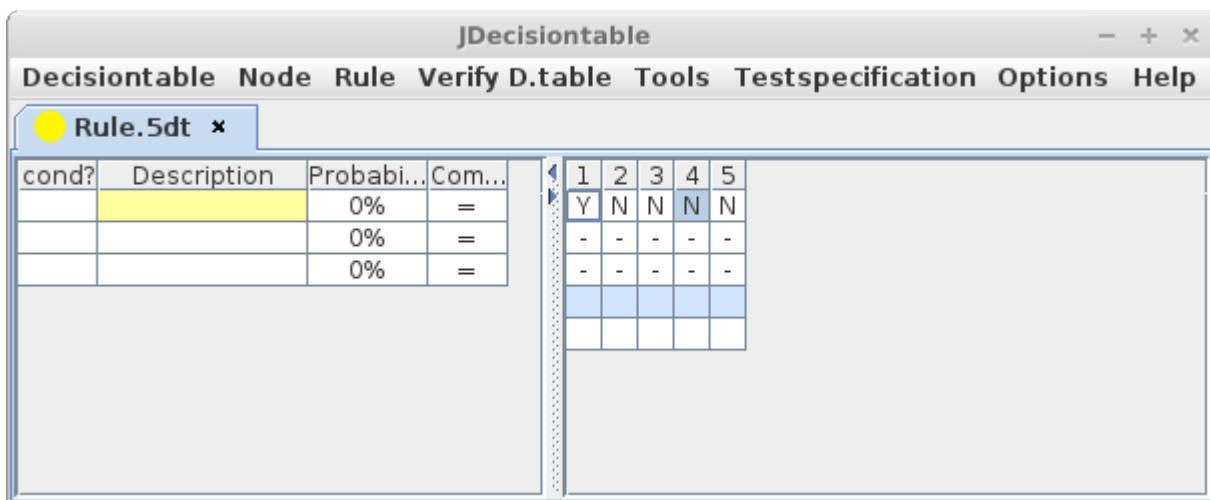


If the source was valid the copy is valid too:

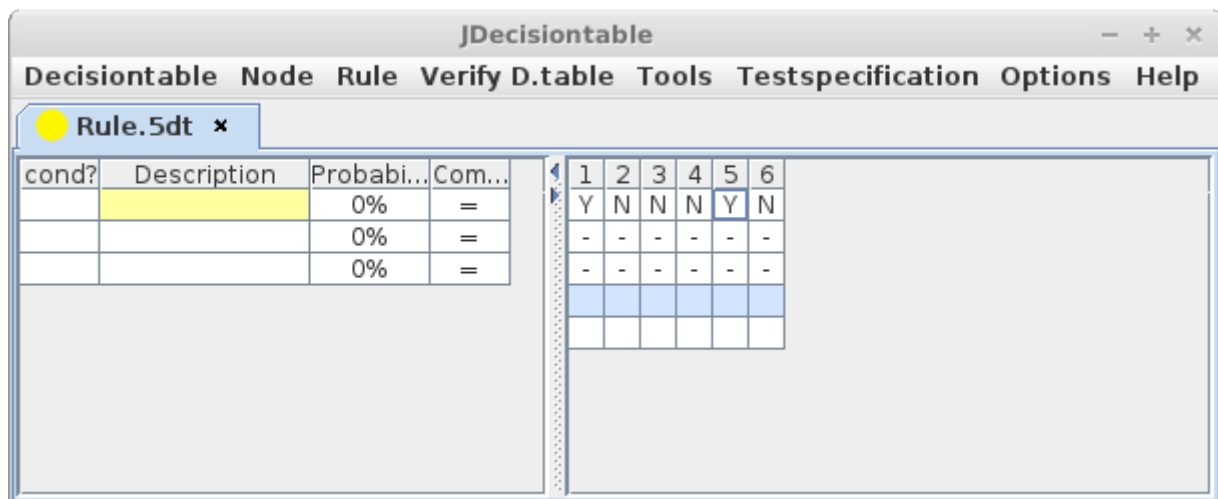
¹¹ Since the first rule of a decision table always starts with a Yes-decision.



As with nodes you may also define source and destination of the copy by selecting rules which are not neighbors:



The copy of the left hand selected cell (rule 1) will be inserted behind the right hand selected cell (this is rule 4, so the copy will be rule 5).



As with nodes, you may also mark a couple of cells (a range) with same result as selecting single cells would have.

Menu Verify D.table

This menu provides tools to determine whether a decision table is valid or not. How this is done see “How to make decision tables” among the help files for this application. There are three checks:

No.	Description	Check passed if ...	Menu item ¹²
1	Check if there is no node with Don't Care – decisions only	no such node was found	Check Nodes_Dontcare
2	Check if all rules are disjunct	all rules are disjunct	Check Rules_Disjunct
3	Check if expected and actual number of rules are equal	expected and actual number of rules are equal	Check Rules_Number
4	Run check 1..3 in the order 1,2,3	checks 1..3 passed	Run All Checks

To determine whether a decision table is valid or not these three checks are executed in a particular order from No. 1 to No. 3 by menu “Run All Checks” (this is check 4 in the table above).

If “Run All Checks” fails (means the decision table is not valid) you may run each of the checks No. 1..3 individually to find out why. It's very useful to start with check 1, if this check passed run check 2, if this check passed run check 3 since the checks 1..3 are build on top of each other¹³.

When a check is completed a message box tells the result.

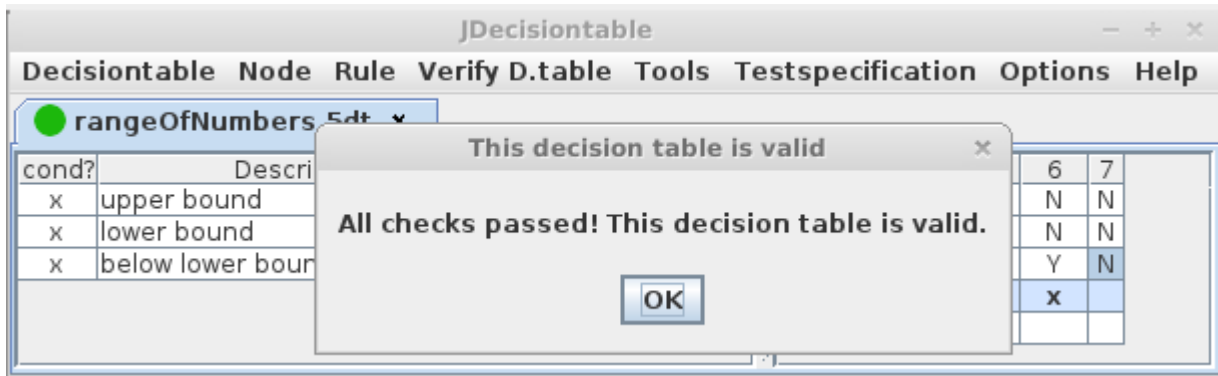
Run All Checks

This menu runs all three checks 1..3 described above to determine whether a decision table is valid or not. It runs exactly the same checks in same order which are used when exporting a decision table to CSV or getting the test specifications. If one check fails the remaining check will not run.

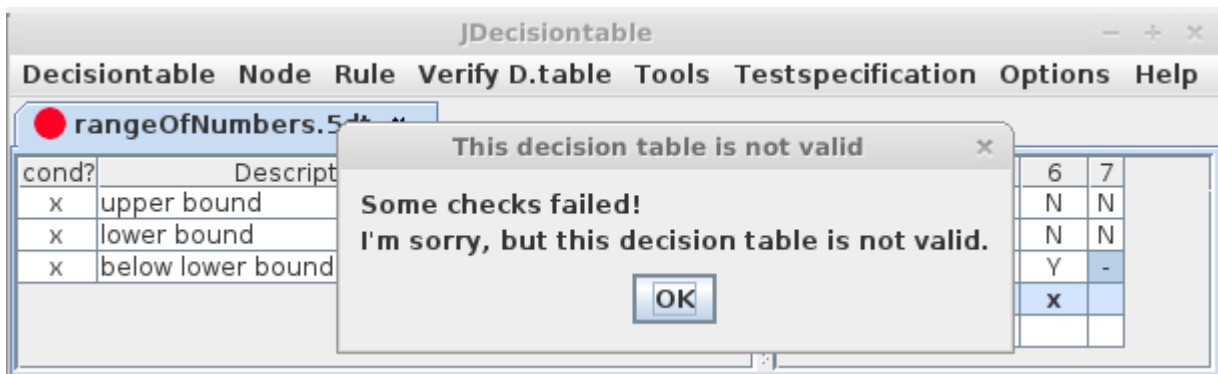
12 Each check has also a name: 1 = Nodes_Dontcare, 2 = Rules_Disjunct, 3 = Rules_Number

13 For example it is possible to create a decision table for which check 3 pass but check 1 fail. But if check 1 to 3 runs one after each other and they all passed there is no chance to pass for an invalid table. This is the reason why checks are executed in this order and why there is a single menu point “Run All Checks”. Please note that these checks examine the right hand table (the rules) only! There are still enough options for mistakes on left hand side!

Example for a valid decision table



Example for an invalid decision table¹⁴



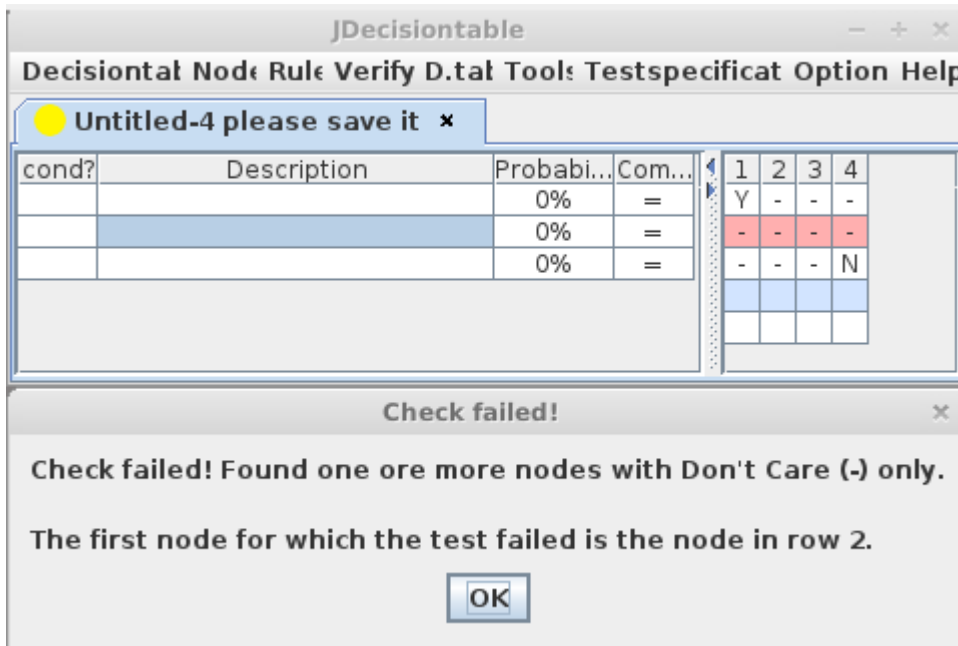
The yellow dot on the tab will turn green if all checks passed and red if not. If you change the decision table this dot will return to yellow. For disabled users there is an alternative icon set using black-on-white-background symbols. Please see chapter “Menu Options” how to switch them on and of. This is the only menu item which changes this icon. Thus if the dot is green it is meant as an optical sign that this decision table is valid.

State	Icon set 1	Icon set 2
Decision table is not valid		
Decision table was changed since last check or was not checked yet		
Decision table is valid		

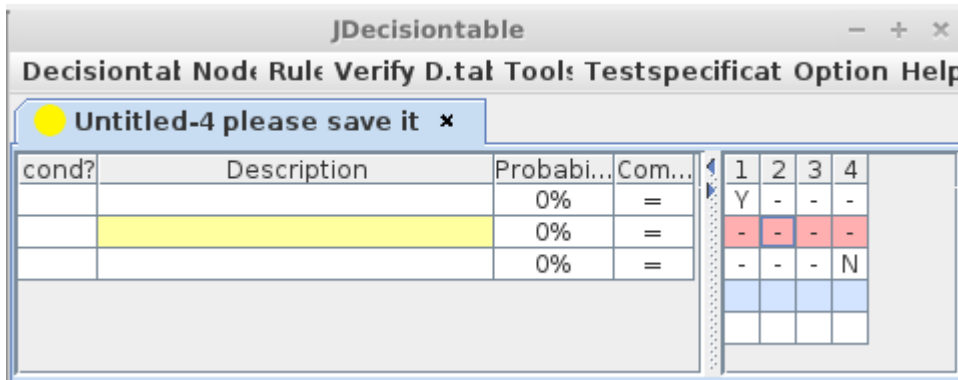
Check Nodes_Dontcare

If any node has only Don't Care decisions this check fails and colours this node in red.

¹⁴ Why did the check fail? Both pictures show almost same decision table. Please watch the last decision in rule 7. Changing it from No to Don't Care let rule 7 not disjunct from rule 6.



You may highlight the description of this node by closing the message box and selecting any cell with red background in the right hand table¹⁵.

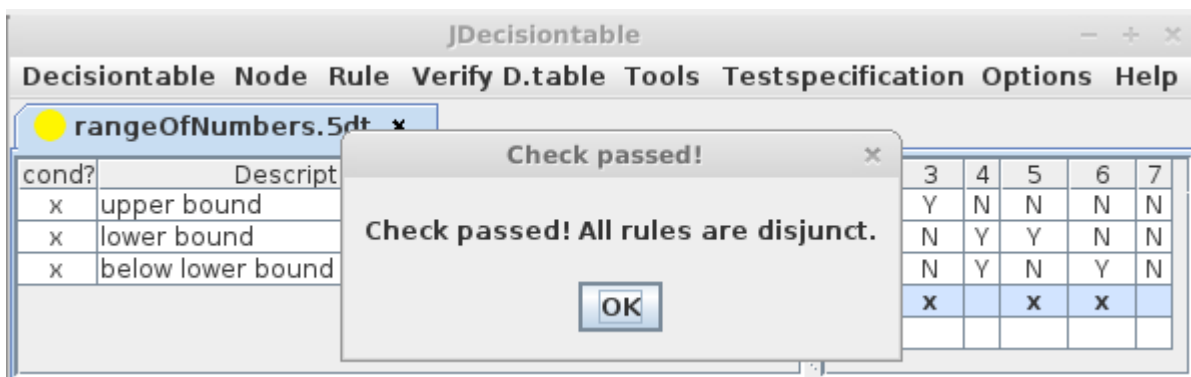


The background colour of the red cells is reset automatically when you run this check again.

Check Rules Disjunct

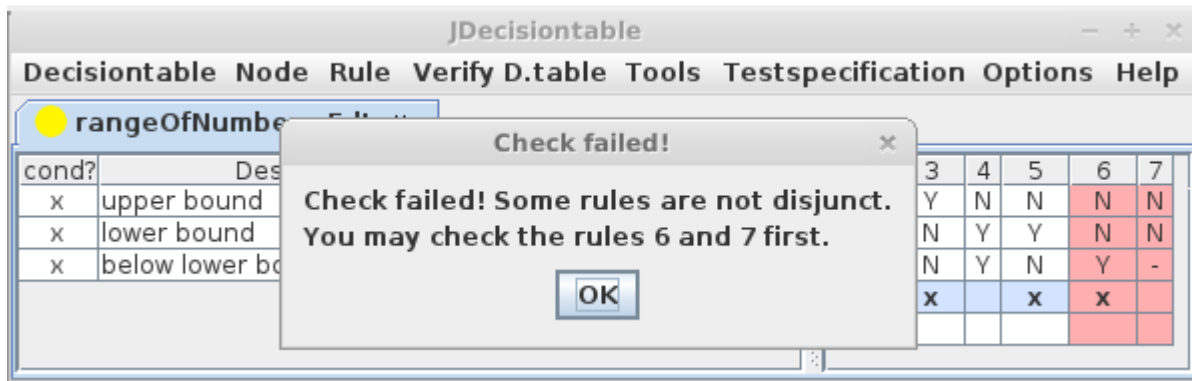
This menu item checks if all rules are disjunct. The document “How to make decision tables” among the help files tells in detail how this check is done. In this application the check will stop if two rules which are not disjunct will found. These two rules get a red background colour.

Example with disjunct rules



¹⁵ You may use Tools → Clear Check Results to reset the background to its default colour.

Example with non-disjunct rules¹⁶

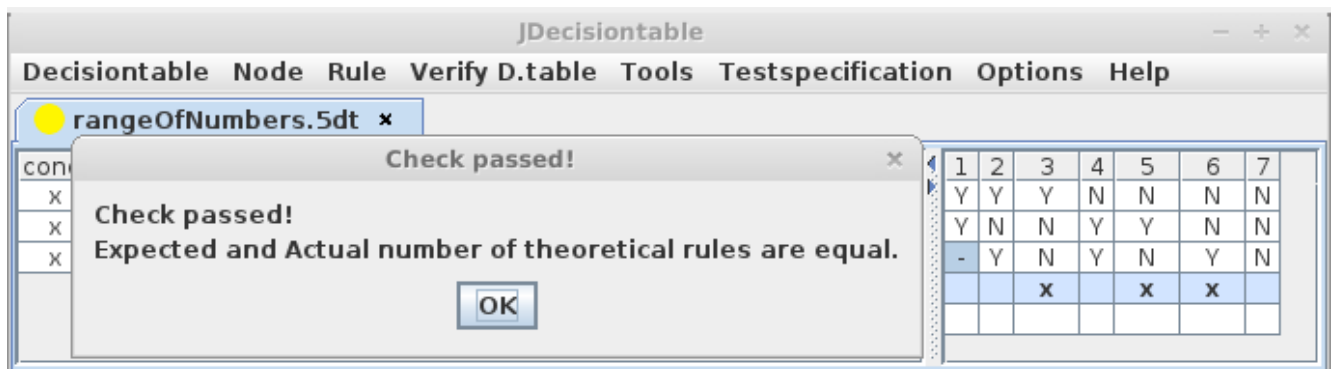


The background colour of the red cells is reset automatically when you run this check again.

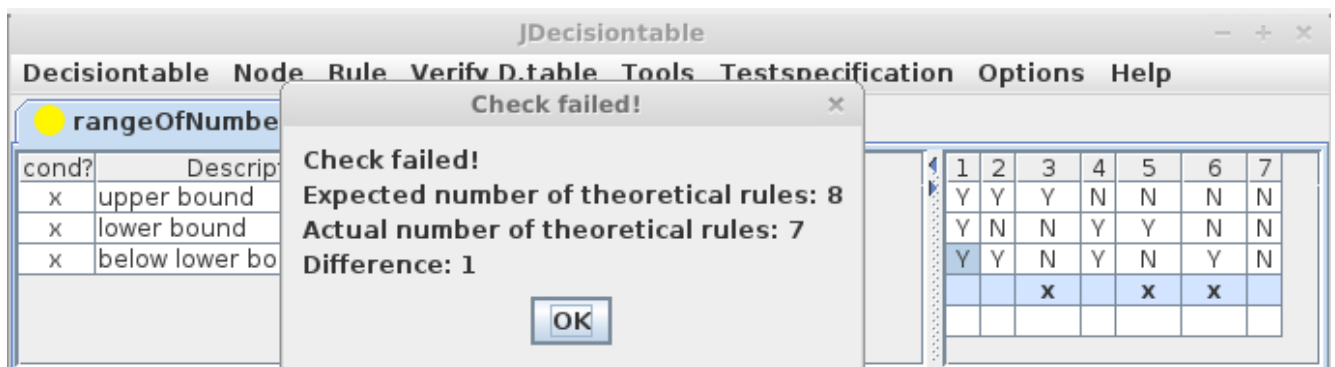
Check Rules_Number

This menu compares the expected and the actual number of theoretical rules. “How to make decision tables” among the help files tells in detail how this check is done.

Example with expected and actual number of theoretical rules matching



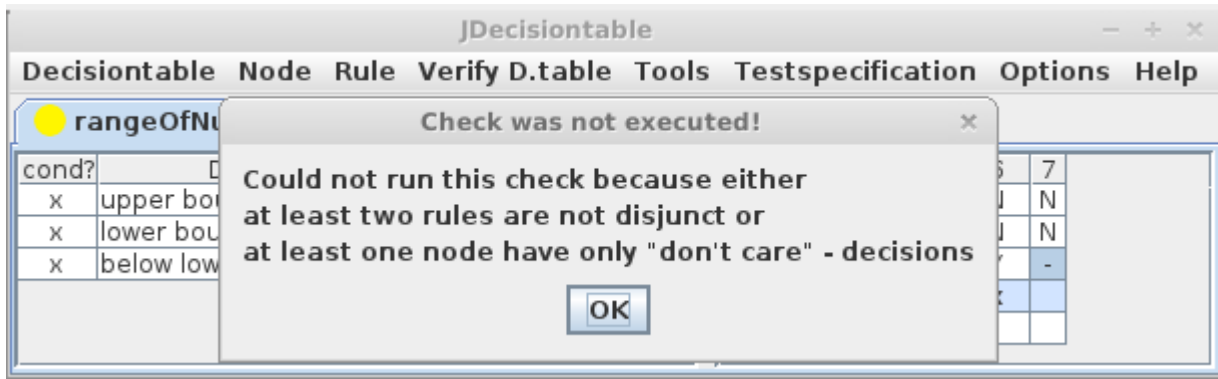
Example with expected and actual number of theoretical rules don't matching¹⁷



This check runs the checks Nodes_Dontcare and Rules_Disjunct before it runs itself. If one of these checks failed this check is not executed. A message box will inform you about what happened.

¹⁶ Both examples show same file but in the example with non-disjunct rules the last decision in rule 7 was changed from Yes-decision to Don't Care-decision.

¹⁷ To make the check fail we changed the last decision in rule 1 from Don't Care to Yes. Since each Don't Care leads to two theoretical rules but Yes or No leads to one theoretical rule (see “How to make decision tables”) the expected and actual number of theoretical rules does not match.



It is recommended to run the checks Nodes_Dontcare and Rules_Disjunct individually to correct the decision table before running this check again. This check will not change the background colour of any cell because this is the job of the particular check.

Show actual num. of th. rules

Shall mean "Show actual number of theoretical Rules". This menu item lets pop up a message box which tells the actual number of theoretical rules. The document "How to make decision tables" among the help files tells in detail how the actual number of theoretical rules is computed.

Show expected num. of th. rules

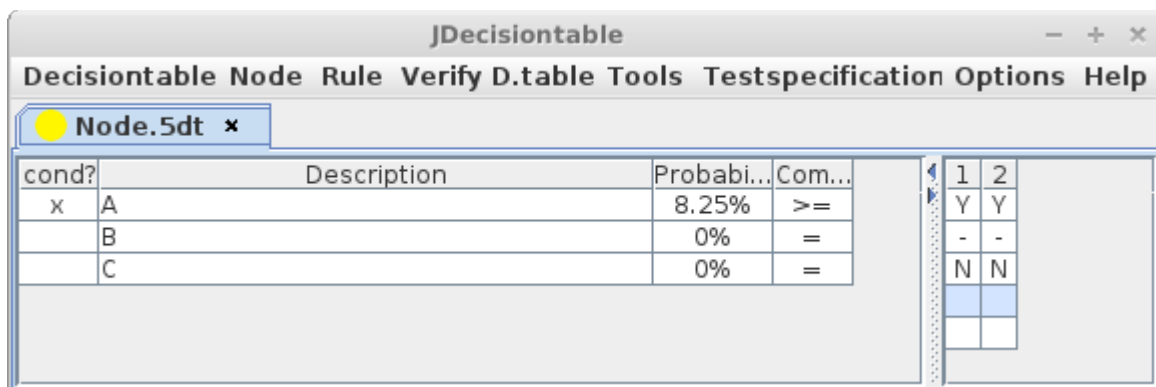
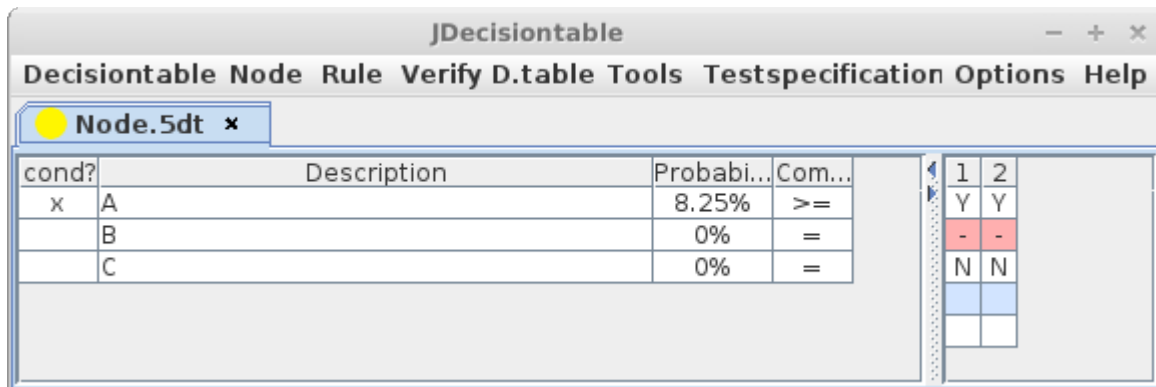
Shall mean "Show expected number of theoretical Rules". This menu item lets pop up a message box which tells the expected number of theoretical rules. The document "How to make decision tables" among the help files tells in detail how the expected number of theoretical rules is computed.

Menu Tools

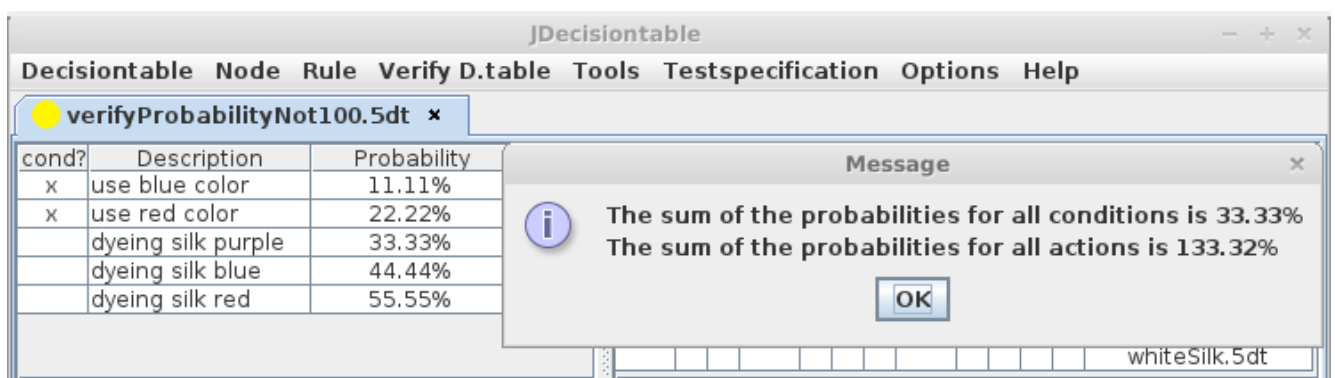
This menu contains some items which provide some interesting functions. Some of these functions are additional checks of the decision table. Although these checks are useful to detect some “smells”¹⁸ or possible improvements.

Clear Check Results

If any check changes the background colour of some cells to mark them, this menu item let you reset the background colour.



Sum Probabilities



As promised this menu item will sum up the probabilities of all conditions and of all actions separately. In detail:

18 Things that are odd but not wrong. “Smells” may cover deeper problems like misunderstandings or improper design. The term “code smell” is common in software development and means a code which works under current circumstances but may break the system under slightly different circumstances.

Conditions:

- use blue color ==> 11.11 %
- use red color ==> 22.22 %

Sum: 33.33 %

Actions:

- dyeing silk purple ==> 33.33 %
- dyeing silk blue ==> 44.44 %
- dyeing silk red ==> 55.55 %

Sum: 133.32 %

This feature may be used for

- usually the sum of probabilities of nodes should sum up each to 100 % and sum of probabilities of rules should sum up to 100 %, so you may use this instrument to check the nodes ==> the left hand side of the decision table¹⁹
(the validation checks in menu “Verify D.table” check the rules ==> the right hand side of the decision table only)
- determining the test cases (each valid rule is a test case) which to run first along the nodes / rules which probability is closest to 100 %
- documenting of business processes in a company (probability of a node: Which condition appears how often?; probability of a rule: Which process runs how often?)
- scientific and market research (i.e. probability of purchases) purposes

Show Nodes w/o Y in Valid Rule

Shall mean “Show Nodes without Yes-decision in at least one Valid Rule”. This check helps to detect a possible serious problem which the validation checks in menu “Verify D.table” are unable to detect: A node has no Yes in any valid rule. Such a decision table can be valid of course²⁰ and it may fit for a particular purpose. But

- Why keep a node which is always ignored or denied?
- What is the use of a condition which never comes true? Or of an action which is never executed?

These questions may help to improve your decision table. If this check detects a node without Yes-decision in at least one valid rule you may

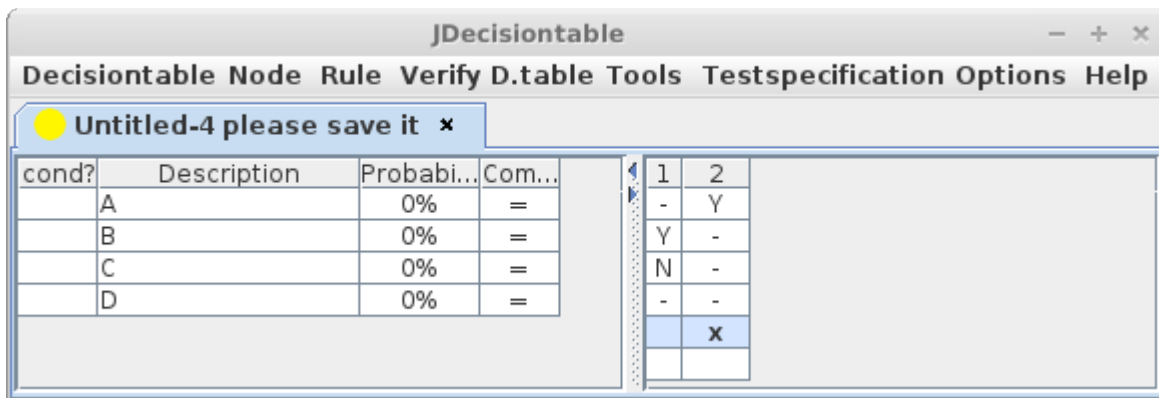
- set a rule to valid
- add a valid rule
- remove the node
- if you find a reason to do so leave the decision table as it is; it may be still a valid decision table

¹⁹ There is also an entry in the project blog

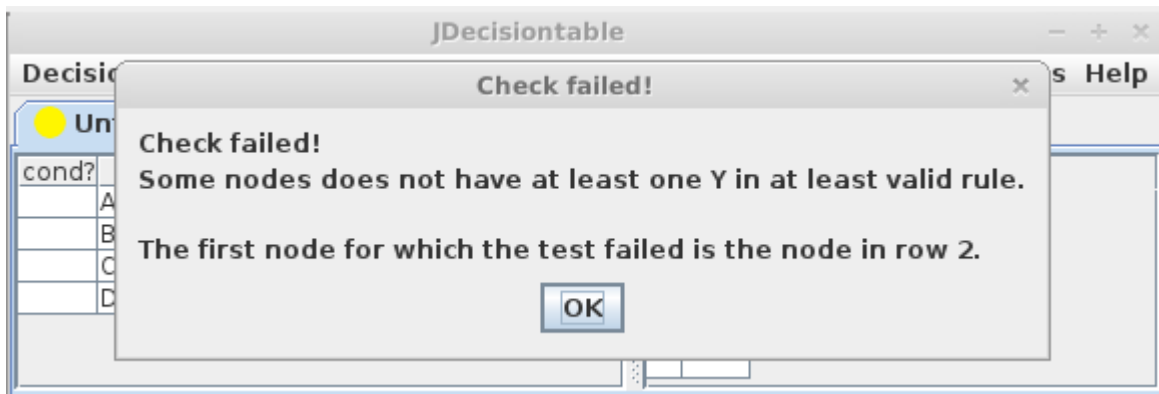
<http://sourceforge.net/p/jdecisiontable/blog/2012/10/use-of-probability-in-decision-tables/>
about probability in decision tables. This point tells briefly what this entry tells in detail.

²⁰ Because validity is determined by definition by a couple of mathematical checks. Validity is not about the if a decision table makes sense.

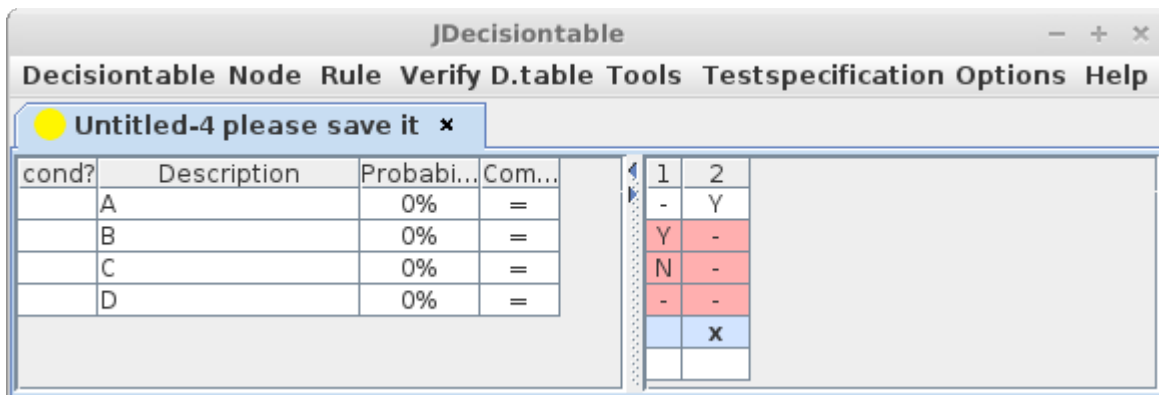
Example: Except node 1, no node has a Yes-decision in a valid rule



The message box gives you a hint there to look first.



All nodes found without Yes-decision in at least one valid rule will be shown with red background colour



The menu item Tools → Clear Check Results resets the red background colour.

Menu Testspecification

This menu let you get test specifications²¹ from decision tables. Test specifications can exported only, there is no way back into the application. There are two different formats available:

- JSON²² for using the test specifications in other software
- CSV²³ for using the test specifications in spreadsheets and documents

Furthermore, getting test specifications may or may not include two functions:

- check if the decision table is valid
- split rules²⁴

The table below shows the eight possible combinations of these four activities.

	1	2	3	4	5	6	7	8
test specification as JSON	Y	Y	Y	Y	N	N	N	N
test specification as CSV	N	N	N	N	Y	Y	Y	Y
check if decision table is valid	Y	Y	N	N	Y	Y	N	N
split rules	Y	N	Y	N	Y	N	Y	N

Only four activities are available (<number> = <menu item>):

- for JSON the two with green background:
 - 1 = Export to JSON +check +split
 - 2 = Export to JSON +check -split
- for CSV the two with yellow background:
 - 5 = Export to CSV +check +split
 - 8 = Export to CSV -check -split

As you see JSON files are always checked if the decision table valid.

21 A test specification is a valid rule in connection with nodes. Please see “How to make decision tables” for details.

22 JSON is also human-readable. There is the plug-in “JSONView” for the web browser Mozilla Firefox <http://jsonview.com>. Just install the plug-in, rename any *.5dt or *.5ts file to *.json and open it with Firefox. See chapter “File Formats”.

23 JSON is a very common data format. There are implementations for many languages (JDecisiontableLib uses Google Gson). Before writing code to parse the csv files please consider to use JSON to read the decision tables or test specifications into your application.

24 Rules which have a Yes-decision for a node with comparison “>=” or “<=” contain two test specifications: One for “=” and one for either “>” or “<”. For a fast overview over a work in progress you do not need to split them but for final test specification this is needed – you would miss a test case otherwise. Please see “How to make decision tables” for details.

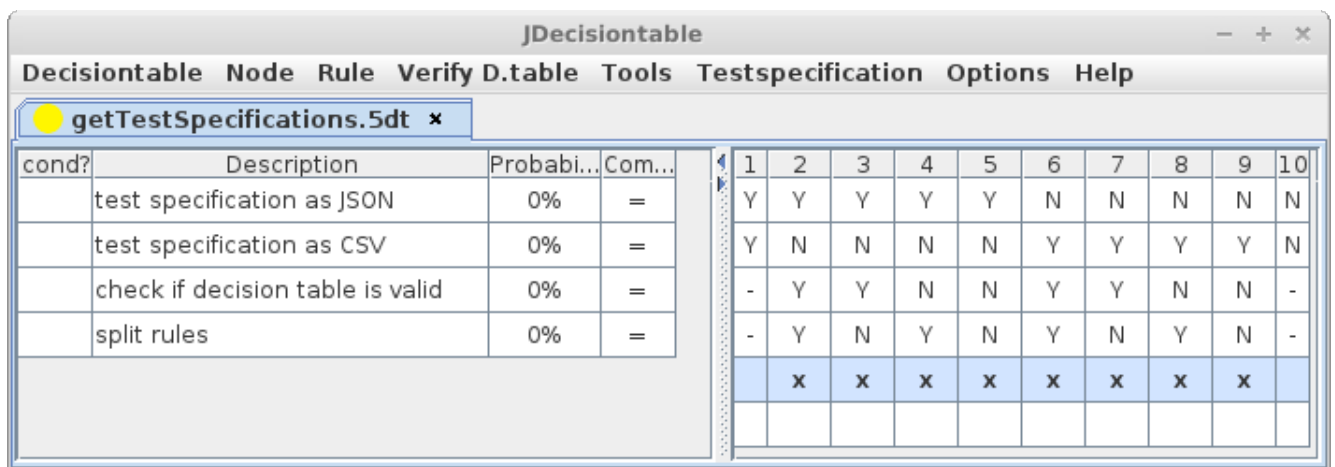
For CSV files, **Export to CSV +check +split** (no. 5) should always be used to get the final test specification for adding test data and execute tests since it makes sure that the test specifications came from always a valid decision table.

Export to CSV -check -split is meant for having a quick self-check while creating a decision table. If the test specification does not show what you intend please check the decision table first. This technique enables you to find mistakes before someone else does.

Menu Options

Make Rows Higher

Make Rows Higher increases the row height like this



Use Icon Set 2

Use Icon Set 2 lets you use an alternative icon set instead of the red/yellow/green dot top left on each tab. See chapter "Menu Verify D.table" → "Run All Checks" for how this icon set looks like and what they mean.

Use English

Use English switches all menus, messages and button labels to the default language (English). You need to select the option and to close and launch the program to bring it to effect.

For what is it good for? This is meant for better collaboration. Imagine another team member calls you. English so at least one's system languages isn't your common language but one/both desktops aren't localised to English. Instead of holding the phone while trying to switch the system language just tell JDecisiontable to "Use English" and share a view to the application via remote desktop connection! It's also useful if you want to do some documentation or blogging for international audit.

It does not change the settings how to format numbers. These are still the same as in your system i.e. if your system's language is German you will still need to input probability using a comma for decimal places – with and without "Use English". It also does not influence how decimal numbers appear in csv files. They will be the same – with this option set or not.

Menu Help

About

Let pop up a dialog window which tells about author, copyright and there to find the original sources.

Help

Let pop up a dialog window which provides a few quick hints.

Command Line

You may open one or more decision tables at launch if you provide them as arguments like

```
JDecisiontable dyeingSilk.5dt getTestSpecifications.5dt
```

All arguments will taken as file names. Arguments known in versions prior to 1.1.0 were removed in favour of an option file.

Option File

JDecisiontable remembers it's settings you made in menu Option. This is done by Java's standard facility for storing options.

The first time when you set an option and close JDecisiontable it will create some folders and files in the home folder of current user. While this place depends on operation systems we will tell it below. The folders and files looks like `de/mgmechanics/jdecisiontable/`.

The is a file `prefs.xml` in each folder. The file used to store the settings for JDecisiontable is `de/mgmechanics/jdecisiontable/prefs.xml`. The other files are unused.

Linux

The file used to store the settings for JDecisiontable is

```
~/ .java/.userPrefs/de/mgmechanics/jdecisiontable/prefs.xml25
```

There are also `prefs.xml` files in the folders `de` and `mgmechanics` but these are not used by JDecisiontable. This is not a fault of JDecisiontable while these folders and files were created by Java. JDecisiontable is not responsible for it. It merely provides key-value-pairs to store. JDecisiontable always writes the actual settings when the application is quitted. If this files and folders does not exist they will be created first time when you set an option and close JDecisiontable.

Windows®

Under Windows®XP® registry keys were created below the registry path:

```
HKEY_CURRENT_USER/Software/JavaSoft/Prefs/de/mgmechanics/jdecisiontable
```

²⁵ “~” is an usual place holder meaning the home folder of the current user i.e. `/home/jane/`

File Formats

This application uses JSON (a data format) to read and write decision tables and to write test specifications. JSON²⁶ is a widely used human-readable²⁷ data format available for many programming languages. It is a text format stored in text files. This should make it easy to use, alter or create decision tables and test specifications with other applications and scripts.

Writing text files:

All text files which JDecisiontable writes are encoded as **UTF-8** without adding a BOM²⁸. According to [RFC3629](http://tools.ietf.org/html/rfc3629) it is possible to save Chinese, Japanese and Korean characters in UTF-8 encoded strings.

Reading text files:

All text files which JDecisiontable reads are expected to be encoded as **UTF-8**. It does not matter if they have a BOM. Successful tests were made to read UTF-8 encoded files having a BOM²⁹ and having no BOM with JDecisiontable version 1.1.1 and at every build of later versions. These tests use German umlauts and special chars to prove JDecisiontable.

26 “JSON” means JavaScript Object Notation. It was originally developed for Javascript and is still very common here. This was one reason to choose it – it integrates well with internet technology.

27 There is the plug-in “JSONView” for the web browser Mozilla Firefox <http://jsonview.com>. Just install the plug-in, rename any *.5dt or *.5ts file to *.json and open it with Firefox.

28 A BOM (= Byte Order Mark) is some extra information at the beginning a UTF-x encoded string. You can see it as bytes EF BB BF before very first char of text in a hex editor. JDecisiontable does not add a BOM at the beginning of UTF-8 encoded files. It relies on Java which does not add it.

29 The test uses Windows® “Editor” editor (notepad.exe) to open a *.5dt file containing characters “ABCxyz123ÄÖÜäöüß@€|μ~¹²³¹/₄¹/₂→{[]}\” and saved as text file with encoding “UTF-8”. “Editor” added a BOM visible as byte EF BB BF at the beginning of the file in a hex editor (Gnome ghex).

By the way: The encoding named “ANSI” by Windows® “Editor” is known as “windows-1252” according to IANA: <http://www.iana.org/assignments/character-sets/character-sets.xml>.

Name	Read/ write	Text/ binary	Description	Suffix
Decision table JSON	read + write	Text	Used to save and read decision tables. This files contain objects of class de.mgmechanics.decisiontablelib.Decisiontable.	5dt
Test specification JSON	write only	Text	Used to store test specifications to open them with other applications. This files contain objects of class de.mgmechanics.decisiontablelib.Testspecification. You can not open these files with JDecisiontable.	5ts
Csv	write only	Text	Used to export decision tables and test specifications suitable to open them as a spreadsheet. Cells are delimited by tab characters (“\t”, hexadecimal 0x09, ASCII code: 09). It was tested with LibreOffice Calc. You can not open these files with this application. The tab character is used as delimiter between cells with intend because you may open any of such csv files in a text editor, copy the content and paste it right into Libre Office Calc or Microsoft® Excel® ³⁰ .	csv
Text	read only	Text	Used to create a decision table using it's content.	txt
Localisation	read only	Text	Used to keep Strings for menus, messages, button labels and even contents for the tables.	json

³⁰ Furthermore, in some European countries the comma is used as decimal separator instead of a period. This might the reason why Microsoft® Excel® uses the colon (;) in the German language version and the comma in the English language version. Using tab character as delimiter avoids confusion.

Hints for translators

JDecisiontable doesn't use the properties files commonly used for Java software. It comes with a unique facility which uses a JSON string stored in UTF-8 encoded text files named **StringResource[_xx][_YY].json**. These are stored in the jar file in the folder **de/mgmechanics/jdecisiontable/**.

It uses a three-step-system.

1. It gets the values for language and country from your system i.e. "de" for language (always lower case) and "DE" for country (always upper case).
2. It reads all values from **StringResource.json**. There are no strings stored somewhere in the source code except program name and version number. Thus, this file keeps all keys which the software uses. The language used in this file is called "default language" and should be English. If the option "Use English" is on the string values from this will be used.
3. It looks for a file named **StringResource_<language>.json** i.e. StringResource_de.json. If found it overrides the values from step before. This file does not need to contain all keys which StringResource.json has. If a key is missing here it takes the value from StringResource.json.
4. At last looks for a file named **StringResource_<language>_<country>.json** i.e. StringResource_de_DE.json. If found it overrides the values from step before. If a key is missing here it takes the value from StringResource_<language>.json or - if the key is missing in this file also - from StringResource.json.

Each JSON contains merely a Hash object (a list of key and values) like:

```
{  
"msgRunAllChecksItem0" : "All checks passed! This decision table is valid.",  
"msgRunAllChecksItem1" : "This decision table is valid"  
}
```

REALLY IMPORTANT: There must be no comma after last entry / before the "}"!

```



```

Its build as “key” : “value”.

The first string is the key which the software uses to find the value. Please do not alter!

After the “:” follows the value: This is the text displayed and in some cases the key to type causing to change the value of a cell. This is to replace with the translated string.

For example, "msgRunAllChecksItem1" : "This decision table is valid"

"msgRunAllChecksItem1" => key

"This decision table is valid" => value

If you need a linefeed just place an backslash “\” instead “” at the and of the line and continue text on next line. “%1\$s” or “%2\$s” are replaced with strings generated at runtime. You may have a look on the value for same key for Default language to see what it is replaced with. You may consider using option “Use English” to get these strings displayed regardless your systems language.

Some values are used to define the key to type causing to change the value of a cell. These keys are named (*.KeyToSet.*). For example:

StringResource.json (English)

"tableKeyToSetDecisionYes" : "y", => User need to type an 'y' to toggle a Yes-Decision

"tableDisplayDecisionYes" : "Y", => right hand table will show an “Y” for a Yes-Decision

StringResource_de.json (German)

"tableKeyToSetDecisionYes" : "j", => User need to type an 'j' to toggle a Yes-Decision

"tableDisplayDecisionYes" : "J", => right hand table will show an “J” for a Yes-Decision (because “ja” means “yes” in German)

StringResource_pl.json (Polish) (to translate)

"tableKeyToSetDecisionYes" : "t", => User need to type an 't' to toggle a Yes-Decision
"tableDisplayDecisionYes" : "T", => right hand table will show an "T" for a Yes-Decision (because "tak" means "yes" in Polish)

StringResource_ru.json (Russian) (to translate)

"tableKeyToSetDecisionYes" : "д", => User need to type an 'д' to toggle a Yes-Decision
"tableDisplayDecisionYes" : "Д", => right hand table will show an "Д" for a Yes-Decision (because "да", pronounced "da" with a very short "a" means "yes" in Russian)

StringResource.json (English)

"tableKeyToSetFlagRuleIsValidTrue" : "x", => User need to type an 'x' to flag a rule as being valid
"tableDisplayFlagRuleIsValidTrue" : "x", => A valid rule will show an 'x' in the field isValid
"tableDisplayFlagRuleIsValidFalse" : "", => A valid rule will show nothing in the field isValid

There is intentionally no key "tableKeyToSetFlagRuleIsValidFalse" because this is set by typing DEL or BACKSPACE key which should same on all systems.

Why there is no undo function and no autosave function too

If you **really** rely on that you don't mess up your decision tables use of any version control system or at least making backups before and during work is **strongly recommended!** It is not recommend to rely on a undo function but it must implemented and tested too. For this reason there is currently no undo function.

Benefit of using an version control system: You can add comment to each version as well as using tags i.e. "these are the decision tables for version x.y of our software". More about see FAQ "I miss an undo function".

There is no autosave function because automatically saving a decision table could write your decision table in a state which you do not want on disk. If you close the decision table it will warn you about loosing data giving you the choice either to save your work or to abandon your changes.