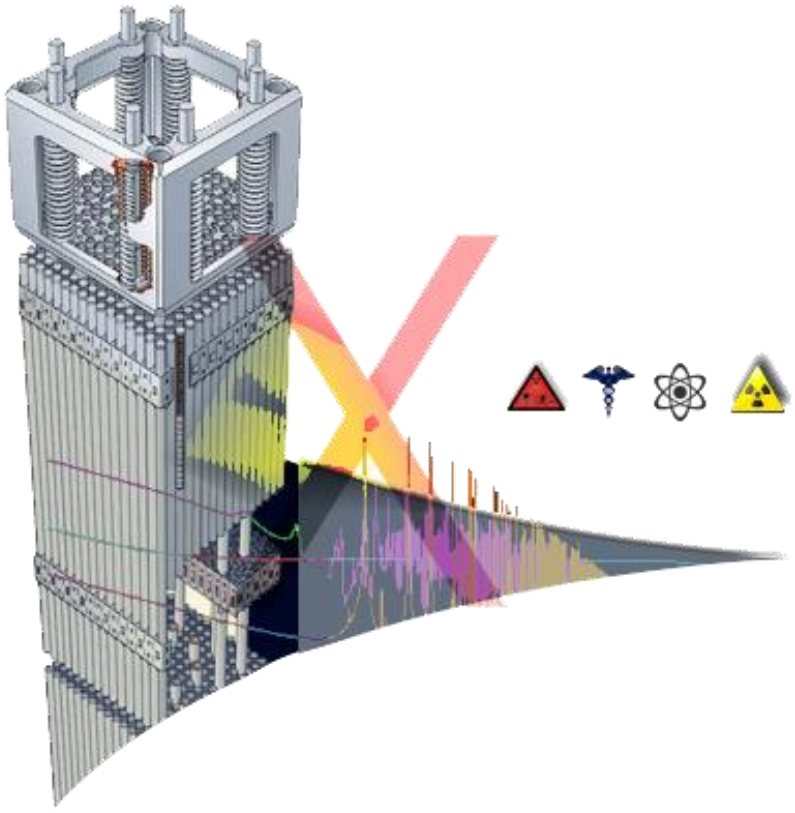
USER GUIDE
THE NUCLEAR EDITOR

Version 0.4 ▪ March 2018



http://www.nucleareditor.com
Harald Breitkreutz ▪ Constructah Internetdienstleistungen

CONTENTS

WHAT IS IT?

The *Nuclear Editor* is an integrated development environment for a variety of nuclear codes, including the most popular MCNP 6.2 and Serpent 2. The main features are:

- Integrated development environment
- Syntax highlighting, structure analysis and input parsing for
  - MCNP 6.2 (alpha)
  - Serpent 2 (in development)
- Intellisense, autocompletion and calltips
- Basic plotting capabilities (experimental)
- Sub-Editors
- Code refactoring
- In-line code execution (experimental)

CODE SUPPORT

The following codes are currently supported by the *Nuclear Editor*[1]:

MCNP 6.2 (alpha stage support)

MCNP® is a general-purpose Monte Carlo N-Particle code that can be used for neutron, photon, electron, or coupled neutron/photon/electron transport, as well as certain other particles. Specific areas of application include radiation protection and dosimetry, radiation shielding, radiography, medical physics, nuclear criticality safety, Detector Design and analysis, nuclear oil well logging, Accelerator target design, Fission and fusion reactor design, decontamination & decommissioning and so forth.

https://mcnp.lanl.gov/

Support for MCNP in the *Nuclear Editor* covers over 95% of the MCNP syntax[2], which should be sufficient for most applications. Nevertheless, the parser code is constantly being improved.

Serpent 2 (support in development)

Serpent is a multi-purpose three-dimensional continuous-energy Monte Carlo particle transport code, developed at VTT Technical Research Centre of Finland, Ltd. Serpent started out as a simplified reactor physics code, but the capabilities of the current development version,

[1] The nuclear codes themselves are not part of the Nuclear Editor software package and need to be obtained separately!

[2] Unsupported features as of now: Certain variations of vertical input, the READ keyword, hybrid geometries, some aspects of magnetic fields, lattices, tallies and sources as well as some minor syntax variations. Furthermore, several deprecated features are not supported. Currently, about 75% of 1500 example and test input files are parsed without errors.

Serpent 2, extend well beyond reactor modelling. The applications can be roughly divided into three categories:

1. Traditional reactor physics applications, including spatial homogenization, criticality calculations, fuel cycle studies, research reactor modelling, validation of deterministic transport codes, etc.

2. Multi-physics simulations, i.e. coupled calculations with thermal hydraulics, CFD and fuel performance codes

3. Neutron and photon transport simulations for radiation dose rate calculations, shielding, fusion research and medical physics

http://montecarlo.vtt.fi/

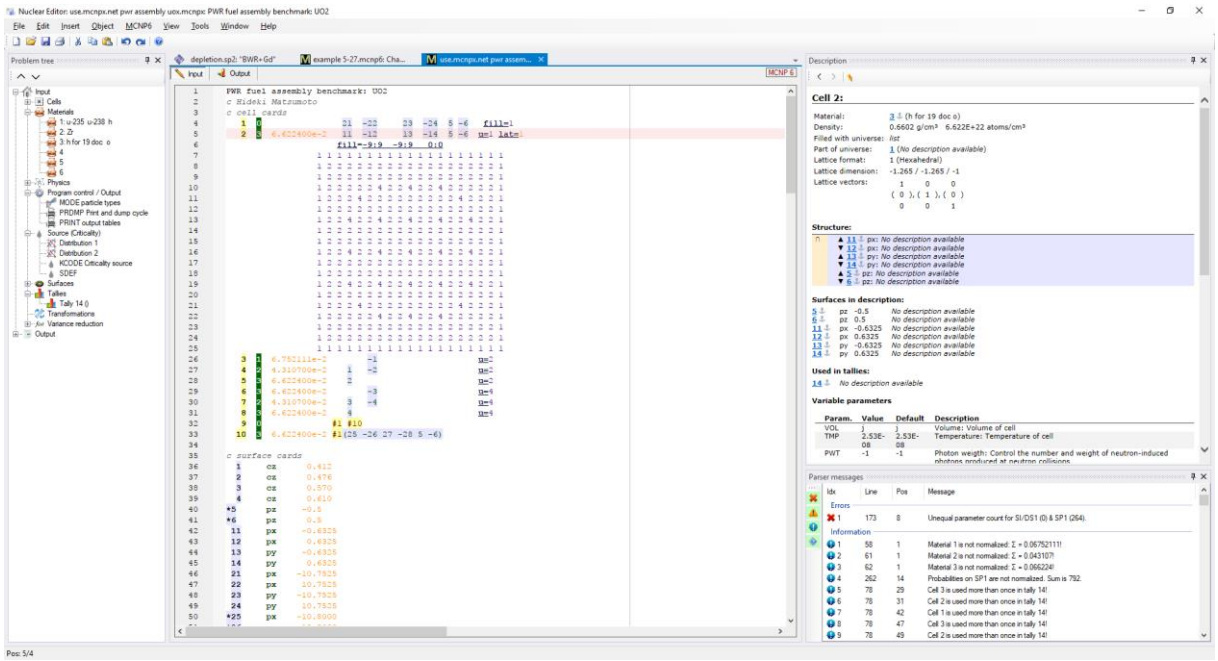
Support for Serpent 2 is currently being implemented, so numerous features are still missing, incomplete or buggy.

Text

The *Nuclear Editor* can display ordinary text files, without any syntax highlighting and structural analysis. This is the only file type the *Nuclear Editor* can display without a valid license.

GENERAL USER INTERFACE

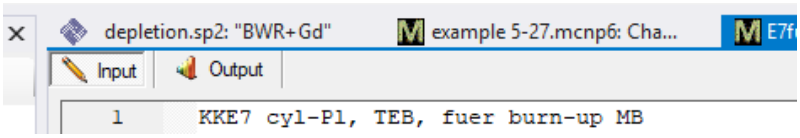
The user interface of the *Nuclear Editor* has been inspired by other IDEs like Microsoft's Visual Studio®.



Basic UI features

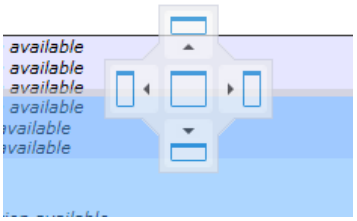
Multiple documents

Multiple documents can be edited simultaneously, even for different codes.

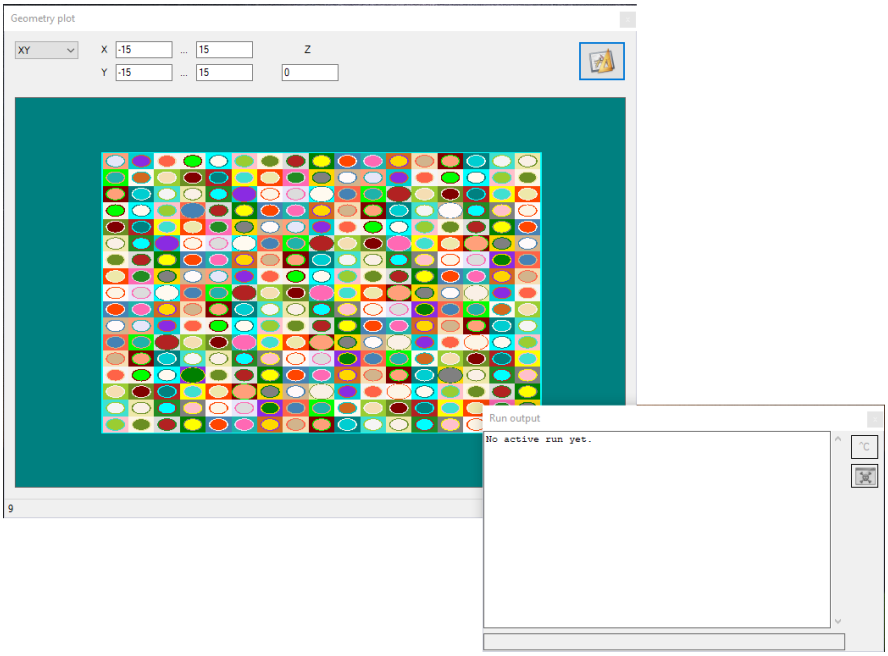


Docking & floating

Windows can be arranged and docked in almost any order and complexity by simple drag & drop. The UI layout is saved automatically when the program is terminated and restored on start-up.



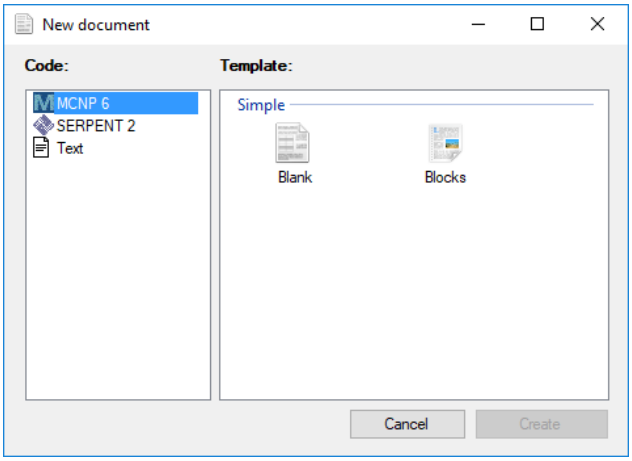
Windows can also float, e.g. on a second monitor, to increase working space. This works for all kind of windows, i.e. tool windows as well as documents.



Basic UI functionality

Basic UI functionality is available from the `File` menu, the toolbar and via shortcuts.

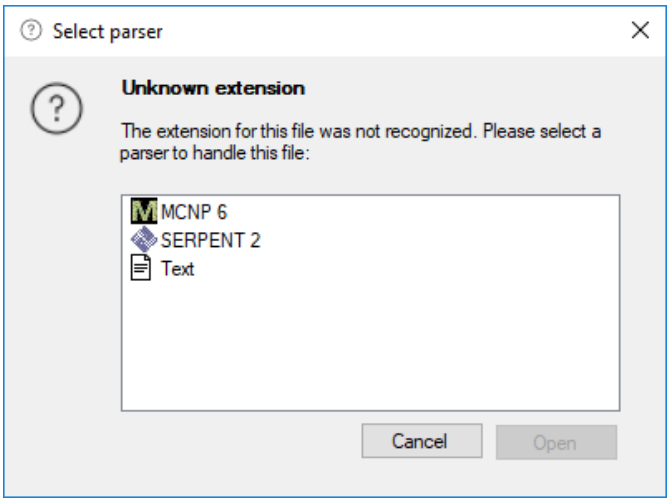
Create **new** file. A dialog window will open that lets you choose from different templates for the various supported codes. Shortcut: `[Ctrl]+[N]`



Open a file. Opens a dialog window to select the file you want to open. The parser is chosen according to the extension of the file. The following extensions are recognized:

MCNP 6: *.m, *.mcnp, *.mcnp6, *.mcnp62, *.mcnpx
Serpent 2: *.s2, *.sp2, *.serpent2
Text: *.txt

If the extension is not recognized, a second window will open that allows you to choose a parser.

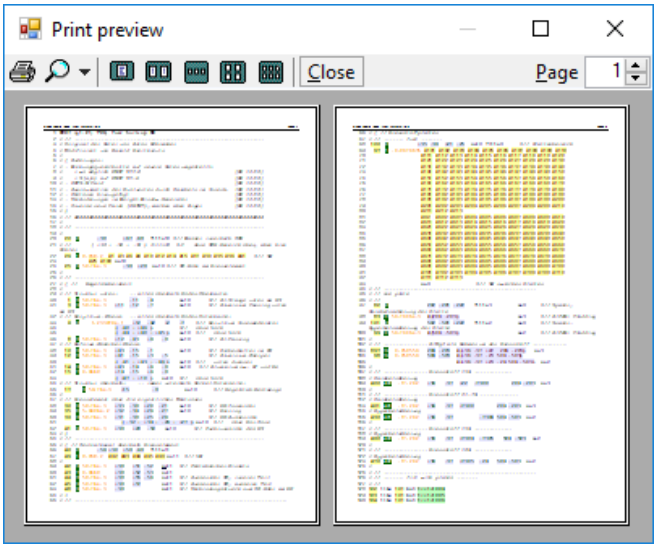


Files can also be opened using drag & drop from the Explorer.

Shortcut: `[Ctrl]+[O]`

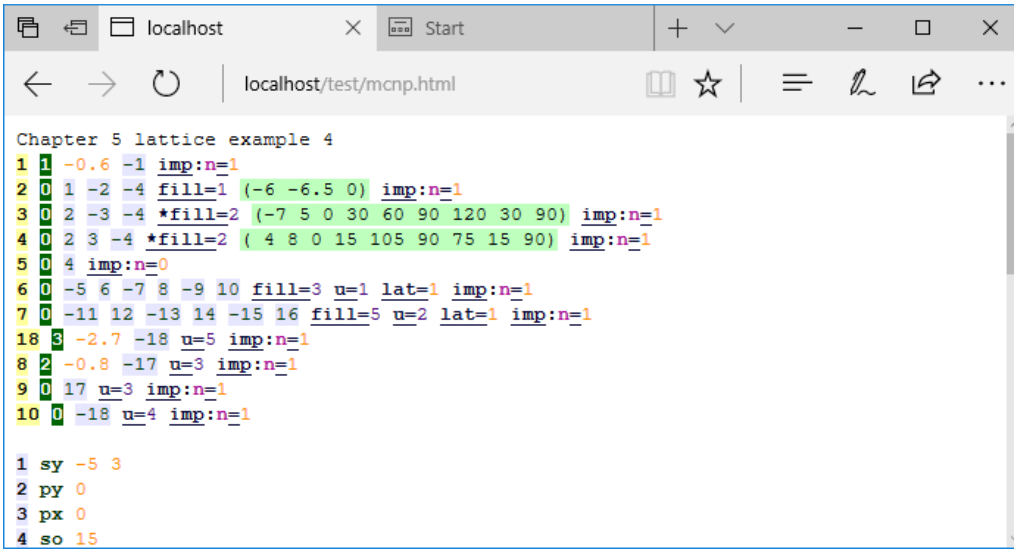
Save the current file. If the file has not been saved before, a dialog will appear that prompts for a filename. If you want to save the file under a different name, choose `File → Save As` from the menu. Shortcut: `[Ctrl]+[S]`

Print the current input, including colouring. If you want to see a preview before printing, choose `File → Print preview` from the menu. Shortcut: `[Ctrl]+[P]`



HTML Export

The *Nuclear Editor* offers HTML export for your documents (`File → Export → HTML`) to view the models in a browser or make them available on a website with the same colouring scheme as in the editor.



Document windows

The document window is your central working place in the *Nuclear Editor*. It contains the input you are working on, and in certain cases, the output from a previous run of the model in a nuclear code.

Your input usually consists of a collection of objects (cell, surface, material, transformation etc.) that are identified by the parser and highlighted. Objects have properties that are specified using either a fixed syntax or via keywords.

Syntax highlighting

Coloring of your input is the central feature of the *Nuclear Editor*. For many nuclear codes, the input consists of barely understandable, long concatenations of numbers and abbreviations. Syntax highlighting visually structures your input, greatly simplifying development.

```
1   point cf-252 fission source in a cylinder of water
2    c     begin cell cards for fixed source sample problem
3    1 1 -1.   -1 -2 3             $ cylinder of water
4    2 0        1:2:-3             $ all space outside the cylinder
5    c       end cell cards for fixed source sample problem
6
7    c    begin surface specifications
8    1 cy  20.                     $ cylinder about the y axis
9    2 py  10.                     $    top plane of water cylinder
10   3 py -10.                     $ bottom plane of water cylinder
11   c      end surface specifications
12
13   c    begin data section
14   mode n p                       $ this is a coupled neutron-photon problem
15   sdef erg=d1 pos=0 0 0 cel=1    $ define a cf-252 pt source at the origin
16   sp1 -3 1.025 2.926             $ use a watt fission spectrum for cf-252
17   imp:n,p  1 0
18   m1   1001 .66667  8016 .33333  $ define h2o using h and o atom fractions
19   mt1 lwtr                       $ use h2o S(a,b) thermal neutron treatment
20   f1:n  1 2 3 (1 2 3)    $ neutron current tally over all sufaces and total
21   f11:p 1 2 3 (1 2 3)    $  photon current tally over all sufaces and total
22   f4:n  1                $ tally the average neutron flux in water cylinder
23   f14:p 1                $ tally the average  photon flux in water cylinder
24   nps  40000             $ run 40000 neutron histories in this calculation
25   print                  $ print everything about the calculation
26   c      end data section
```

The actual coloring scheme depends on the code for that you are writing a model. However, we have tried to keep it as much as possible consistent across the supported codes:

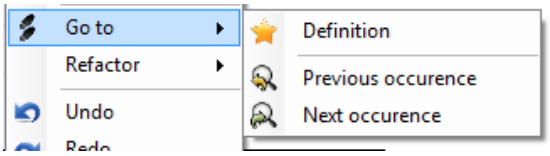
1	cell (letter color is altered if it is an excluded cell)
1	surface (letter color is altered for included / excluded surfaces in cell definition)
1	universe, lattice, lattice element
1	material, mixture (or anything else material-related)
1	isotope, cross-section
1	tally, detector, mesh… - anything that will give a result in Monte Carlo codes
1	source-related, distribution
n	particle-type related
e	object / card property (or input cards that could also be an object property)
p	other input card
1	ordinary number
3	modifier, special number
p	ordinary text
c	comment
1	syntax error

Additional coloring is used for more code-specific features and varies among the supported languages. Usually, pre-defined groups of numbers are also visually grouped by means of a frame surrounding them (e.g. for vectors).

The coloring scheme is currently not customizable, however, that is something on the (long) list of things we want to implement.

Quick navigation

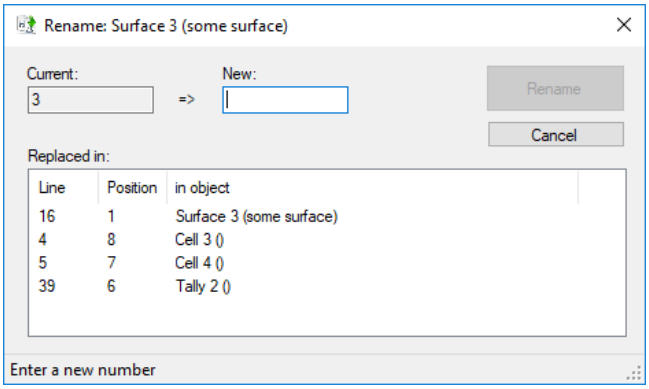
Jump quickly to the definition of an object by pressing `[F12]`, go to the next (`[Alt]+[→]`) or previous occurrence (`[Alt]+[←]`). Can also be accessed from the context menu (right click) of the object.



Refactoring

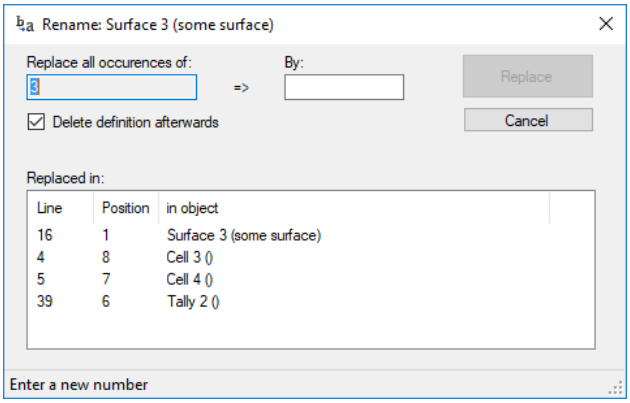
Renaming

Use this to rename an object and update all references to it in the document in on single step. The editor will check for collisions. Useful to clean up your input, e.g. by introducing new, consistent numbering schemes for your objects. Shortcut: `[F2]`



Replace and delete

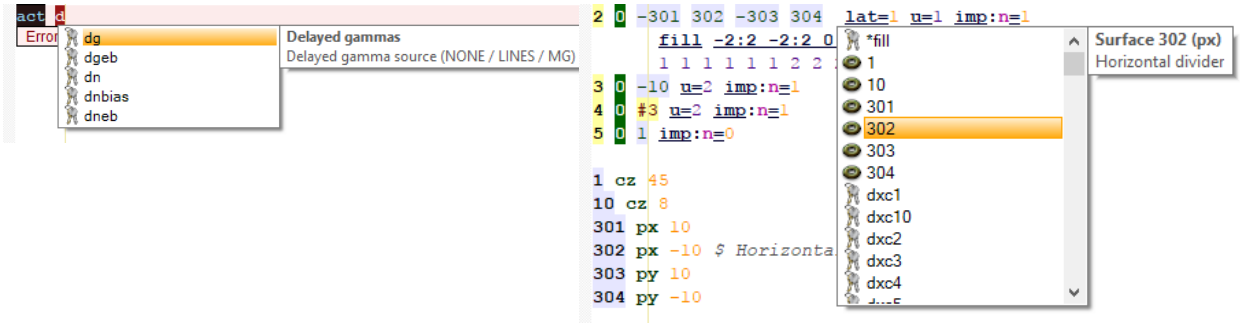
This will replace all occurrences of an object (cell, surface, material, transformation etc.) by a different object and optionally deletes the definition of the original object. Use this to remove duplicates from you model easily.



Intellisense / Autocomplete

Support: MCNP 6

Basic autocompletion capabilities exist to facilitate input of keywords and numbers for cells, surfaces etc. To show the menu, enter `[Space]` and the press `[Ctrl]+[Space]`.

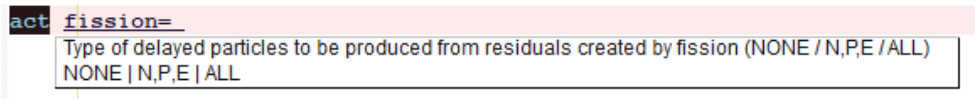


Autocomplete is currently only implemented for MCNP 6. The capability is not available everywhere and currently not guaranteed to lead to 100% syntactically correct results.

Tooltips

Support: MCNP 6

A tooltips is a small helper window to assist with complex input syntax. The tooltip is automatically displayed when a character is entered on the keyboard in a cursor location where tooltip information is available.

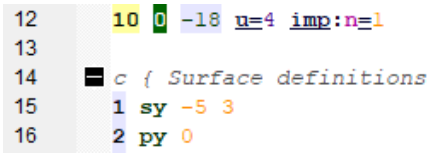


Input folding

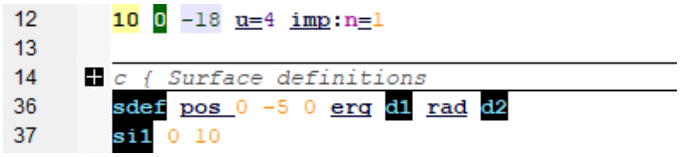
You can hide certain parts of your input in the document window. Add a comment line at the beginning of the block and an opening curly parenthesis. At the end of the block, add another comment line and a closing curly parenthesis. For MCNP 6, you would add the following two lines:

```
c {
...
c }
```

Next to the upper line, a black minus sign will appear. Click on it to hide the input portion between the parentheses.



Once folded, the upper most line of the block will remain visible and a black plus sign will appear. Click on it to expand the block again.

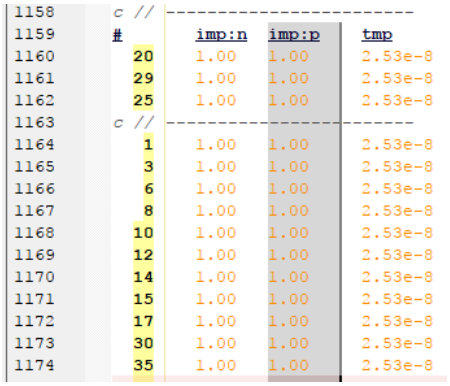


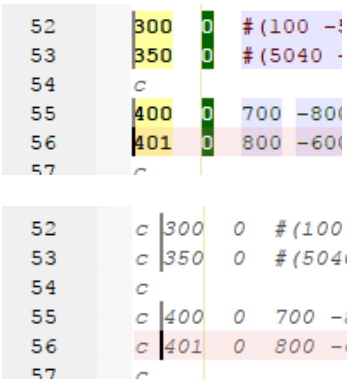
Inline parser messages

You can display messages of the parser (see → *Parser messages*) directly in your input, i.e. below the line where the parser incident occurred. This feature can be enabled or disabled for every message type in the `View` menu.

Rectangular and multiple selection

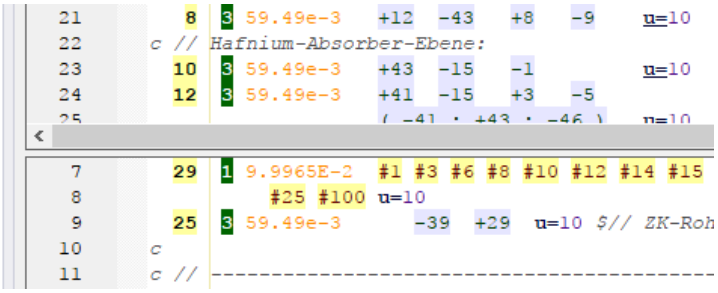
By holding `[Alt]+[Shift]` and using the arrow keys, either a multiline-cursor or a rectangular selection can be made. Alternatively, you can hold `[Alt]` and select an area using the left mouse button. Using [Ctrl] and the mouse, multiple selections can be made at the same time. This feature is especially useful if you want to comment out several lines at the same time or to remove columns of input (provided your input is properly formatted).





Split-View

It is possible to split an editor window in either the horizontal or the vertical direction into two parts using the `Window → Split` menu.



Split-Views are slightly more performance-intensive than single views. To turn a split-view back into a normal window, drag the splitter completely to one of the sides.

Standard editor functionality

Of course, the document window provides all standard text editor functionality, via the main menu, the context menu, the toolbar and shortcuts:

	Cut	`[Ctrl]+[X]`
	Copy	`[Ctrl]+[C]`
	Paste	`[Ctrl]+[V]`
	Undo	`[Ctrl]+[Z]`
	Redo	`[Ctrl]+[Y]`

Tool windows

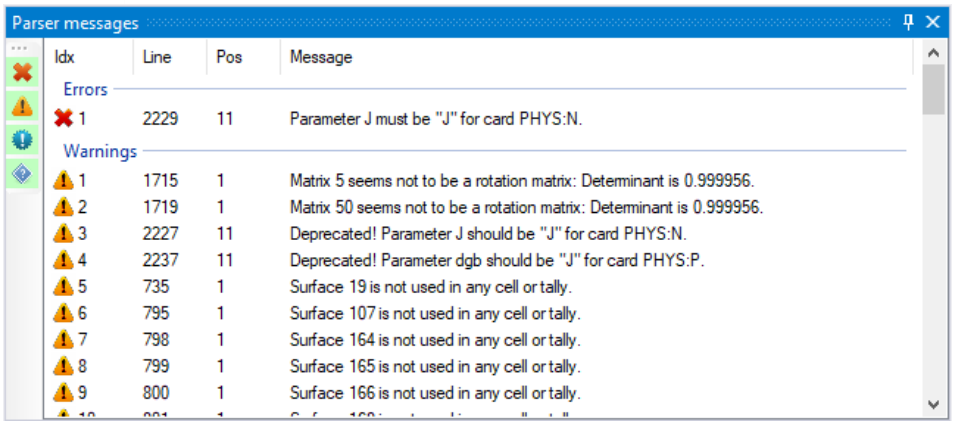
The following helper windows are available from the `View` menu of the Nuclear Editor:

Parser messages

While the parser analyses the structure of your input, it will look out for syntax errors and other things it considers suspicious. These findings are displayed in the parser messages window and grouped according to four different categories (error, warning, info, unsupported). The buttons on the left side of the window allow you to filter the message groups that are displayed:

- Turn on/off error messages
- Turn on/off warnings
- Turn on/off informational messages
- Turn on/off messages regarding unsupported features and depreciated syntax

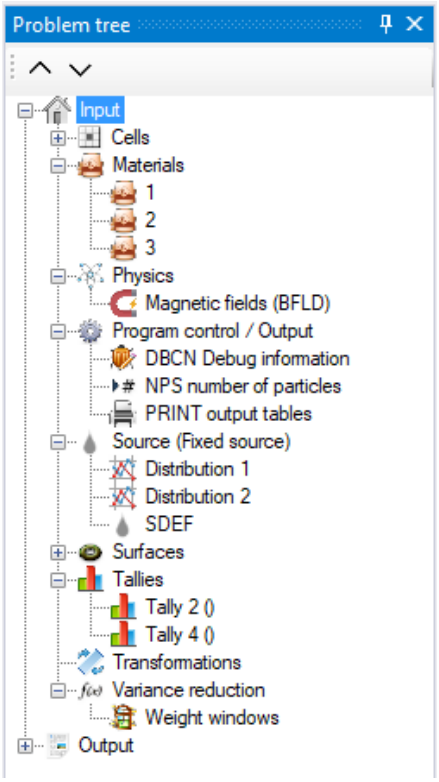
Enabled message filters are shown in green, disabled in red.



Double-click on a message in the list to navigate the cursor in the document window to the origin of the message.

Problem tree

The problem tree is a navigation tool for your input. Basically, it displays the result of the structural analysis the parser performed on your input.



The problem tree groups the object in different categories. Actual grouping depends on the code you are developing for. MCNP6 input objects, for example, are grouped according to cells, surfaces, materials, tallies and transformations. Further cards are grouped in physics, program control and output related, source definition, variance reduction etc.

Navigation in the tree is like in a normal directory tree. Double click on an item to set the cursor in the document window to the definition of that object.

Click on ⌄ to expand the full tree. Click on ⌃ to collapse the whole tree.

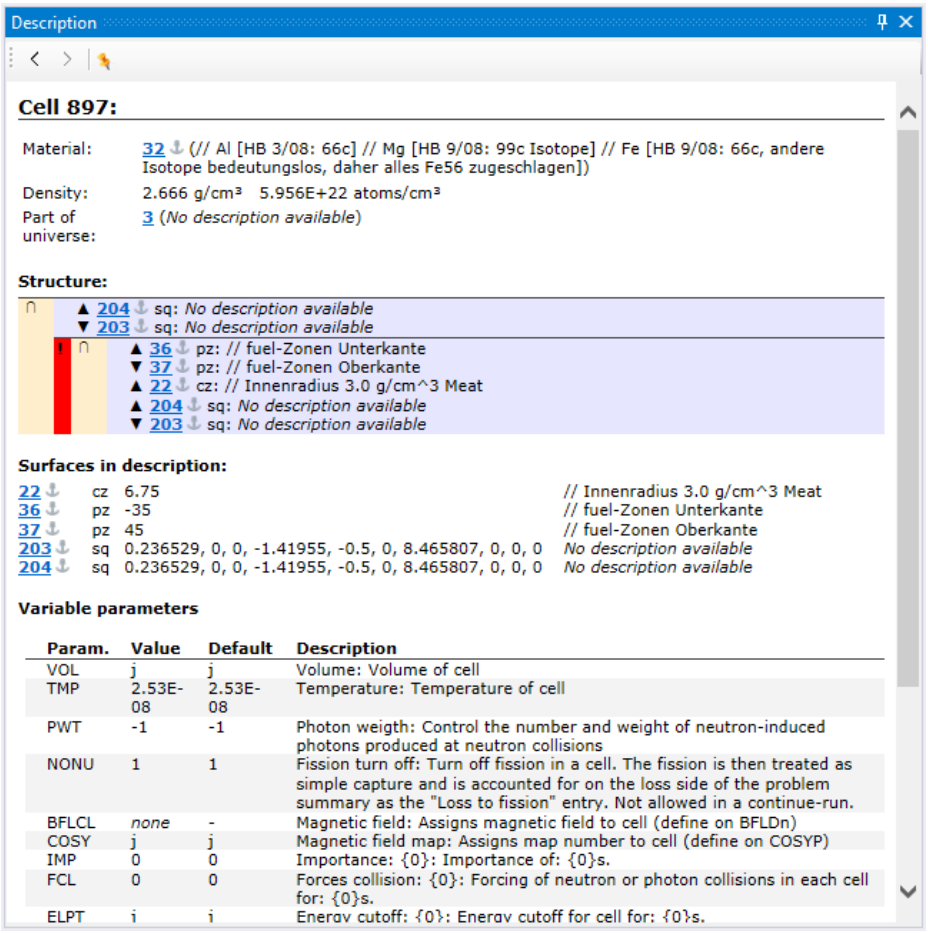
Description (up to two windows)

Especially for documents with higher complexity, the description window will be one of your central working tools. When the cursor in the document window is placed on an object, the description window will summarize the information of the object. For example, it will show the structure of a cell, i.e. how it is composed from surfaces, its fill material, its parameters etc.

Basically, the description window is a mini-browser displaying the information in form of a website.

- Click on a blue link and the window will display the information that object.
- Click on the anchor icon next to a link (⚓) to set the cursor in the document window to the definition of the object the link leads to.

- Click on < to navigate backwards and > to navigate forward in the history.
- Activate the needle icon ( ) to pin the window content. The description will no longer change when you move the cursor in the document, and even if you switch between different documents.



Optionally, the description window can display the source code of the object. This has to be enabled in the Options-window.

You can have one or two description windows:

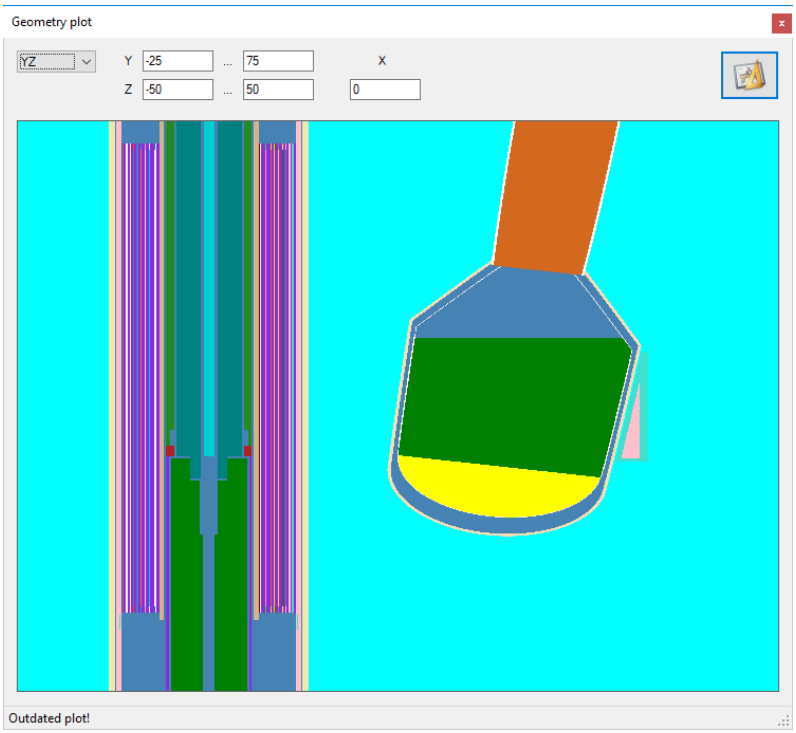
- If you have only one, the window will always show the description of the object on which the cursor is positioned.
- If you have two, the primary window will show the description of the object that governs the line in which the cursor is positioned, while the secondary window will show the description of the object the cursor is placed on. For example, if a cell is defined of surfaces and the cursor is placed on the surface number, then the primary window will show the description of the cell, while the secondary will show the description of the surface.

Geometry plot (experimental)

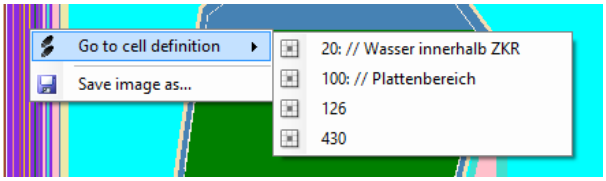
Support: MCNP 6

The *Nuclear Editor* contains experimental, basic plotting capabilities. It makes use of multi-threading and is pretty fast, however it does not yet support all geometries (macrobodies, certain surface types and certain lattice types). The plotter currently only supports simple XY, XZ and YZ views. Surfaces will not be plotted, only cells.

The plot will not automatically update when you edit your model due to the computational expense. Click the button in the upper right corner to update the plot.



If you right click in the plot window, the *Nuclear Editor* will determine the nested cell path to the innermost cell at the point you clicked and offer you the possibility to navigate to any cell in the path. Left click in the plot to display the cell path in the status bar.

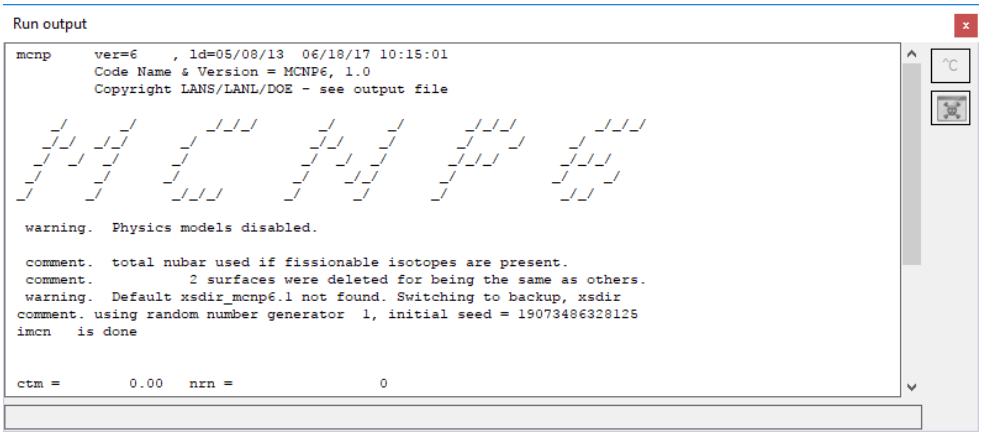


You can also save the plotted image using the context menu.

Run output (experimental)

Support: MCNP 6

As an experimental capability, the *Nuclear Editor* supports running of codes in-line. This needs to be configured via the → Options dialog. If so, the program will be executed in the background and its output captured. The results will be displayed in the run output window.

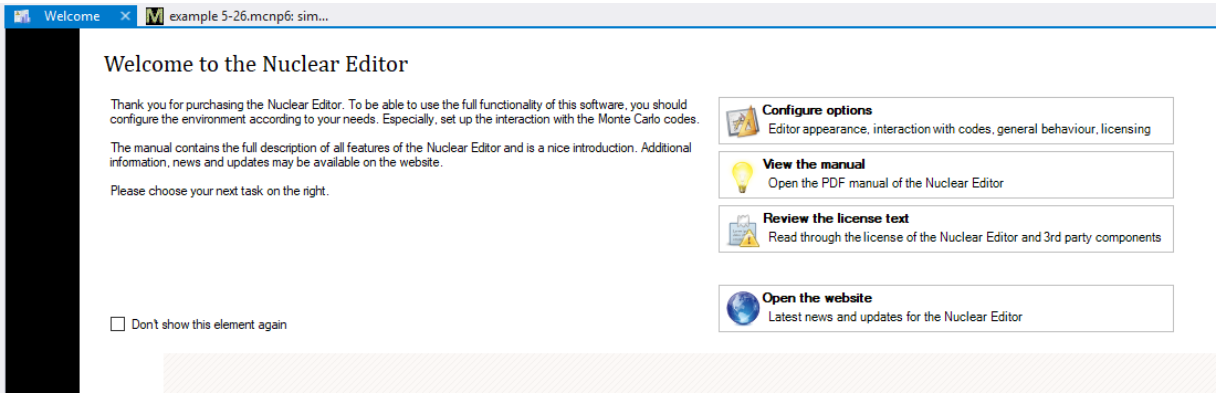


You can interact with the code through the two buttons on the left. Clicking the ^C icon will send `[Ctrl]+[C]` to the code. Clicking on will kill the code. Using the command line below the output, you can send ordinary commands to the code.

This capability is experimental and might be removed in future versions if it doesn't proof to be useful.

Welcome window

A welcome window will be displayed on first start-up and if there is no document open. The welcome window fetches online content with news and updates about the Nuclear Editor and provides help on the first steps with the Nuclear Editor. You can remove the latter by checking the "*Don't show this element again*" box.



You can always display the window again by selecting the `Tools → Welcome screen` menu.

CODE-SPECIFIC FEATURES

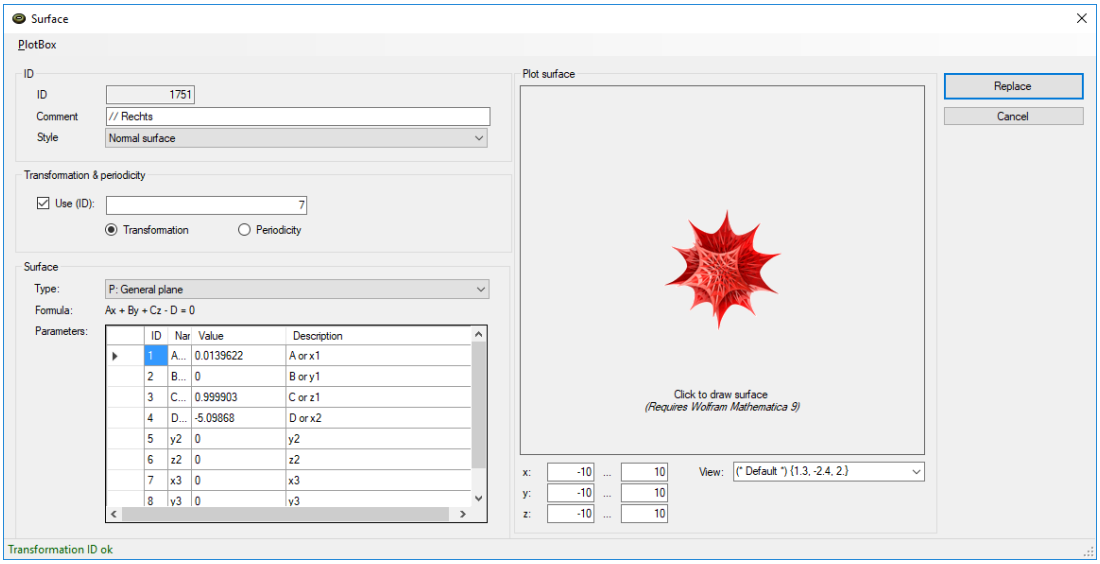
Code-specific editors

Surface editor

Support: MCNP 6

Surfaces in most nuclear codes are defined by implicit equations. The surface editor is a convenient way to input the parameters of such a surface. The various pre-defined surfaces

including their equations are built into the editor, as well as description of the parameters. Furthermore, you can define reflectivity, periodicity and transformation for the surface.

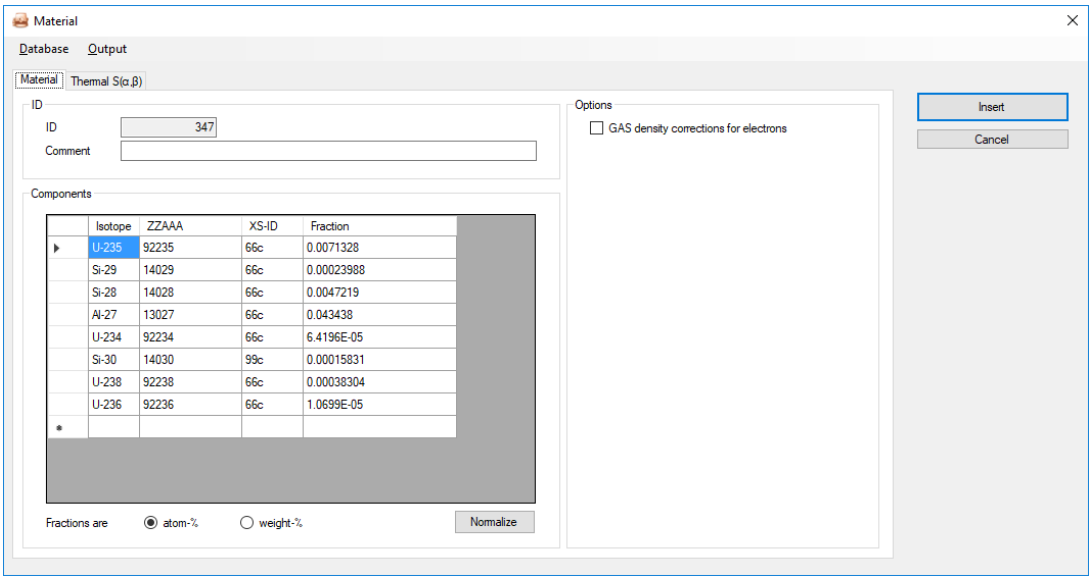


If Wolfram Mathematica (version 9 or better) is installed on the same machine, the actual surface can be plotted in 3D.[3]

Material editor (in development)

Support: MCNP 6

The material editor is a convenient way to add or edit materials in your problem.

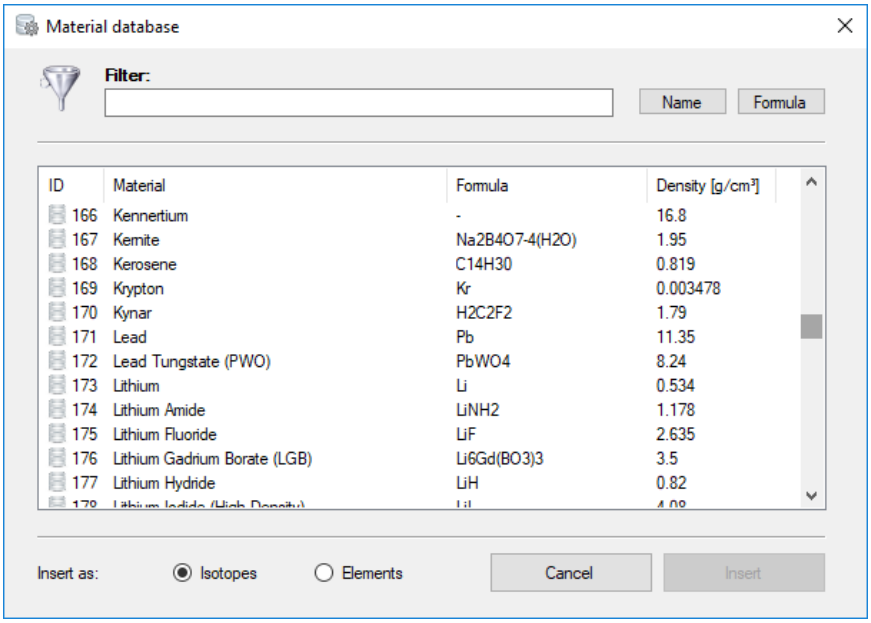


The material editor is still in development and does not support all material-related features of MCNP6.

A large material database with several hundred pre-defined materials can be accessed via the `Database → Load` menu directly from the material editor. It covers most standard materials.

[3] We plan to remove the Mathematica dependency in future versions.

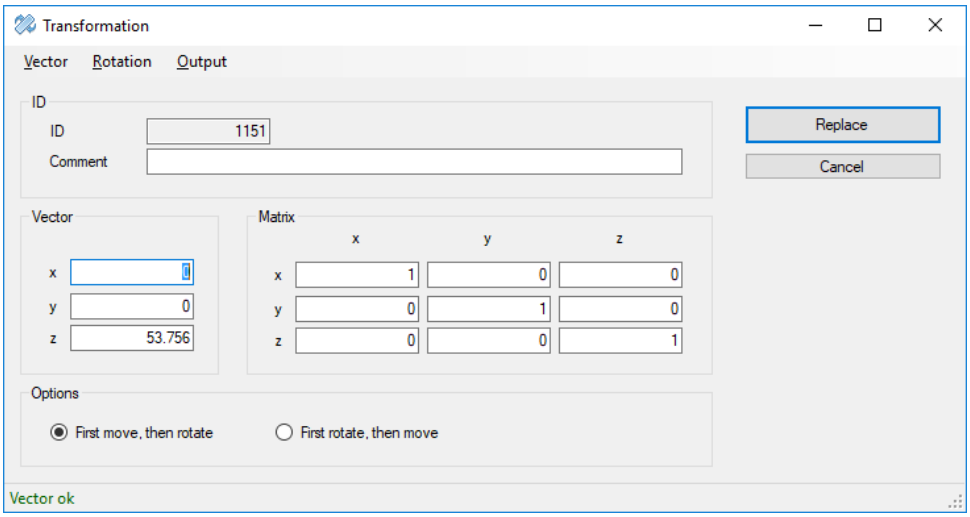
The list can be filtered either by name or formula of material with the respective input fields. If an elemental cross section is available (like ZZ000.xx), you can choose to use these, or break the element down into its isotopes.



Transformation editor

Support: MCNP6

Transformations can easily be defined using the Transformation editor. You can either input the translation vector and the rotation matrix directly, or choose from various pre-defined parameter sets.



MCNP6.2

MCNP6® was the first supported code in the *Nuclear Editor*, therefore support for this language is the most advanced.

Code execution

If MCNP6 is installed on the same machine and properly configured via the → Options dialog, it can be executed directly from the *Nuclear Editor*. The MCNP6 code is not shipped together with the editor and needs to be obtained separately from RSICC[4].

MCNP6 can either be executed in an external window (default) or in-line in the *Nuclear Editor* (experimental). The latter option allows for a tighter interaction between the *Nuclear Editor* and MCNP6.

The following run-modes are available from the `MCNP6` menu or via shortcuts:

Initialize only: Runs MCNP6 with the IX option, i.e. the run will be terminated after the initialization. This option is useful for checking for errors and warnings that have not been captured by the *Nuclear Editor*'s internal parser.[5]

Shortcut: `[⇧]+[F5]`

Run: Runs a full MCNP6 calculation. The Nuclear Editor will detect automatically whether the current problem is a fixed-source problem or a criticality calculation and will pass the source point file accordingly to the command line.

Shortcut: `[F5]`

Plot geom.: Runs MCNP6 with the IP option to plot the current geometry. As this utilizes the MCNP6 plotter, it requires an X-Server to be installed on the machine. If the plotter fails, make sure that the `DISPLAY` variable has been set accordingly in the options.

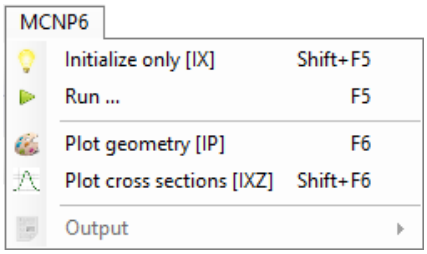
Shortcut: `[F6]`

Plot XS: Runs MCNP6 with the IXZ option to plot cross sections. This utilizes the MCNP6 plotter, too.

Shortcut: `[⇧]+[F6]`

[4] https://rsicc.ornl.gov/Catalog.aspx?c=M

[5] The MCNP6 parser of the Nuclear Editor is completely independent of the MCNP6 internal parser, therefore error detection works somewhat different. While the parser tries to detect as many errors as possible, it is very likely that it will not find all errors that the MCNP6 parser will find.

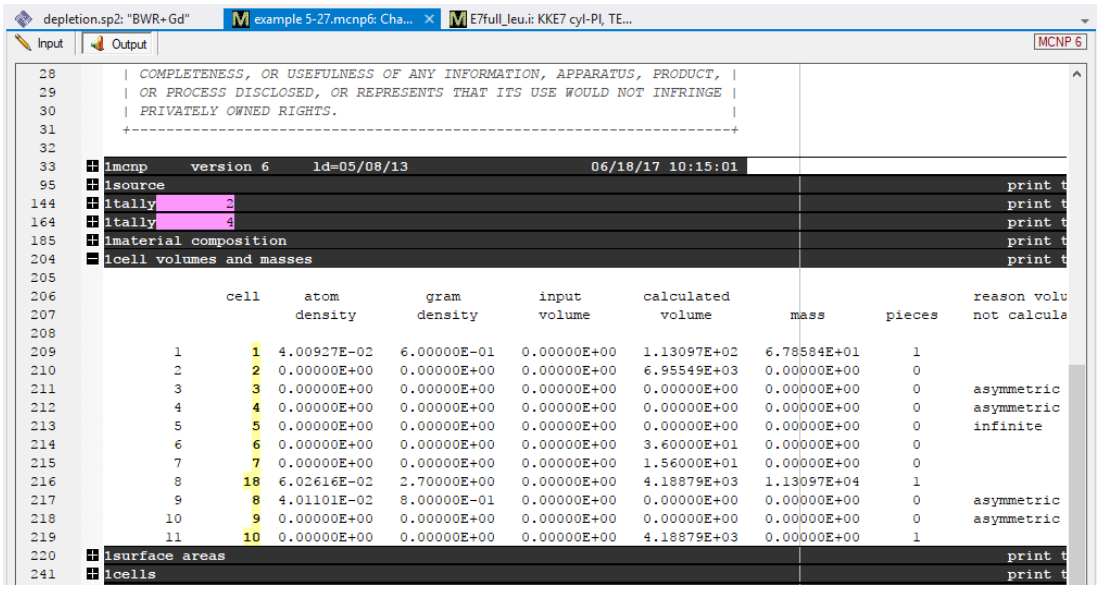


After every run, the output file will be copied and an existing output file will be automatically overwritten. The output file will afterwards be displayed in the output analysis window (see section below) for further analysis.

Several options to configure the way MCNP6 is executed by the Nuclear Editor can be configured via the → Options dialog.

Output analysis (in development)

After an MCNP run is finished, or if you open a file where a corresponding output file exists, the Nuclear Editor will perform an analysis of the output. Since this capability is still in early development, not much will happen besides some syntax highlighting and the ability to collapse certain portions of the output.



To view the output, click on the Output button above the document. To go back to input view, click on Input.

Import from output file

An MCNP output file quotes the model that was used to generate the output. You can extract the model from an output file to a new file by choosing `File → Import → MCNP6 result file` from the menu.

Serpent 2

No Serpent 2 specific features are currently available as support for this code is still in early development.

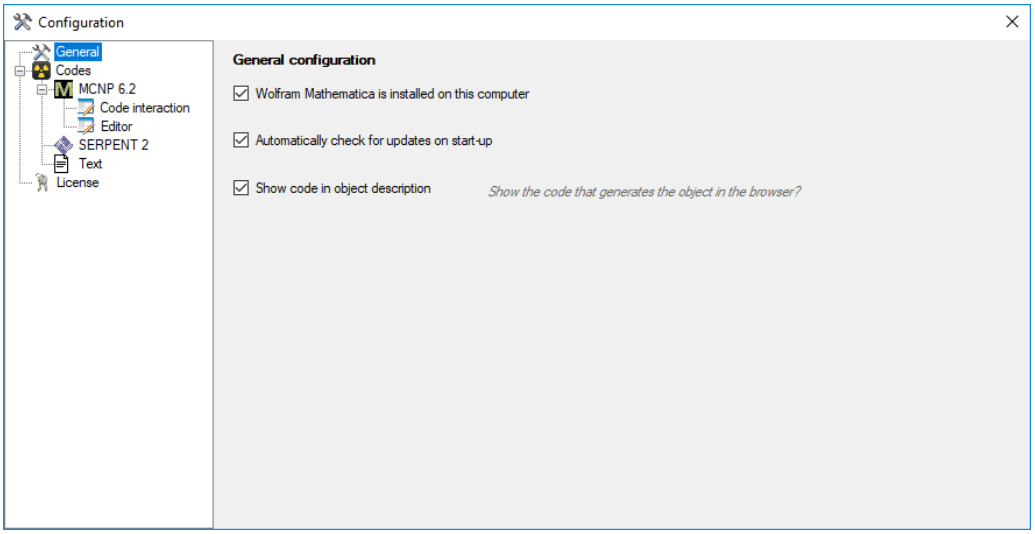
OTHER TOOLS

Options / Configuration

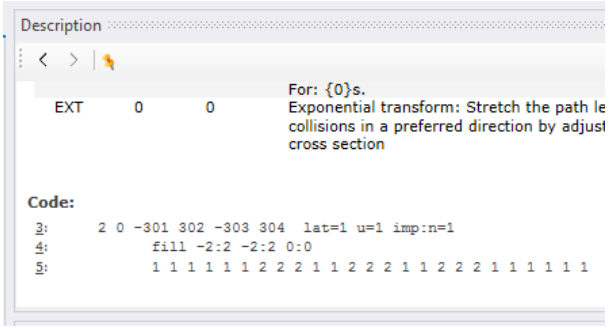
To configure the behaviour of the *Nuclear Editor*, use the `Tools → Options` menu.

General

Configuration options that apply to everything in the editor.

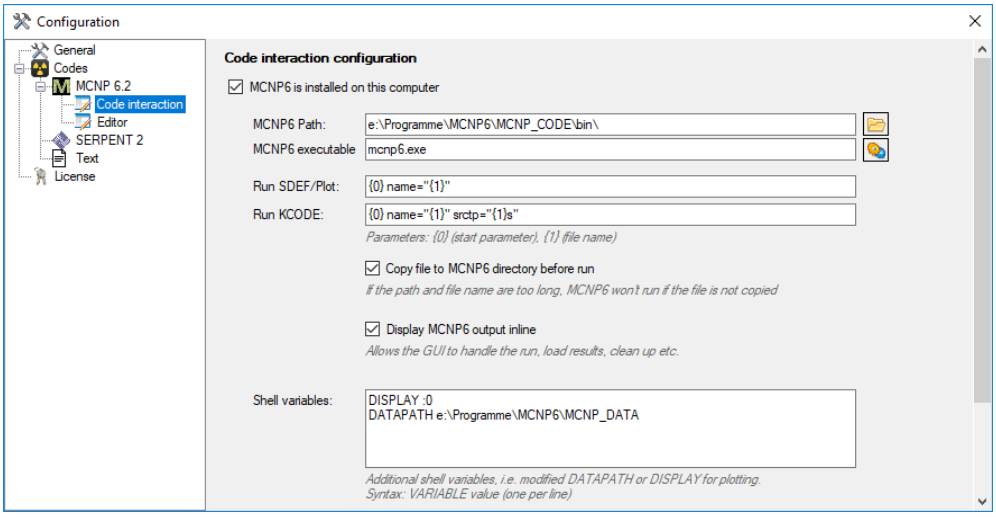


- *Wolfram Mathematica® is installed on this computer*: Tick the box if this tool is installed on the computer. It will enable additional functionality for surface plotting.
- *Automatically check for updates on start-up*: Tick the box if the *Nuclear Editor* should go online at start-up of the software to check for a newer version of the software. Make sure your firewall is configured to allow internet access for the *Nuclear Editor* to http://www.nuclearediter.com. You can always check for updates manually using the `Help → Check for updates` menu.
- *Show code in object description*: In the description window, the source code that created an object can be shown if this option is selected.



MCNP 6.2 Code interaction

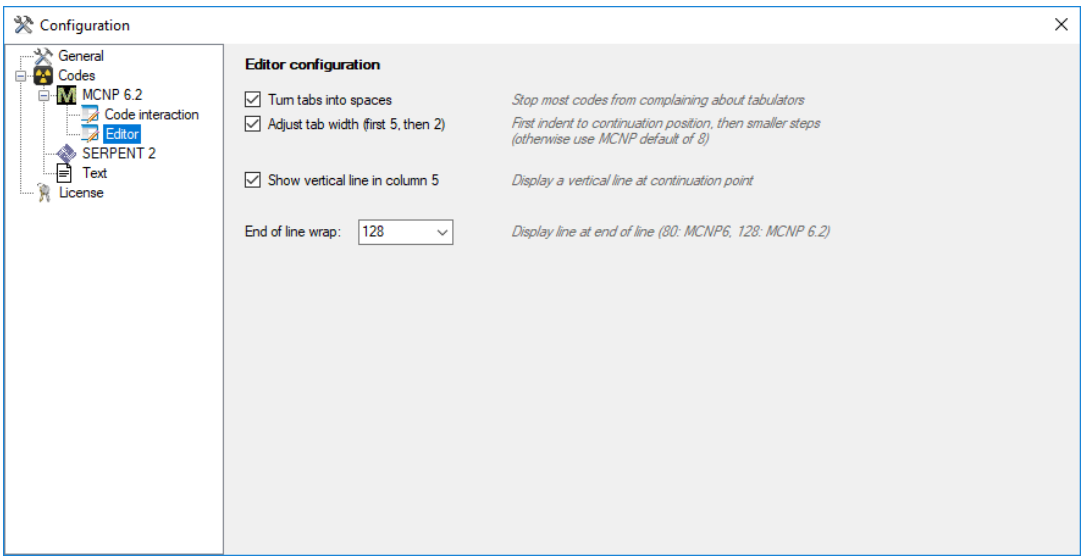
Configuration specific for MCNP 6.2 code interaction and execution.



- *MCNP6® is installed on this computer*: Tick the box if MCNP6 is installed on the computer. If not, no features that require code execution will be available. It is recommended to have MCNP6 installed on the computer if you work with this code in the Nuclear Editor.
- *MCNP6 path*: Enter the directory where MCNP6 is installed, including a trailing back-slash. You can also select the directory via a dialog by clicking on the icon on the right.
- *MCNP6 executable*: Enter the name of the MCNP6 executable (default: `mcnp6.exe`) or select it via the icon on the right. No path information is required, but the executable must reside in the directory selected as MCNP6 path.
- *Run SDEF/Plot*: Enter the command to run MCNP6 for fixed source calculations or to plot the geometry. Use "{0}" as placeholder for the parameters (like "ip" for plotting) and "{1}" as placeholder for the filename. Default: `{0} name="{1}"`
- *Run KCODE:* Enter the command to run MCNP6 for criticality calculations. Default: `{0} name="{1}" srctp="{1}s"`
- *Copy file to MCNP6 directory before run:* Check to copy the input file from its actual location to the MCNP6 directory before the code is executed. Avoids trouble with long directory and file names (recommended).
- *Display MCNP6 output inline:* Enable the experimental inline-run functionality of the *Nuclear Editor* for MCNP6.
- *Shall veriables:* Set additional shell variables for MCNP6. Usually, you want to set `DISPLAY` for plotting (most of the time `DISPLAY :0`) and the `DATAPATH` to the cross section files.
- *MCNP6 manual:* Set the path to the MCNP6 manual file (PDF).
- *MCNP6 xsdir:* Set the path to the MCNP6 xsdir file to allow the *Nuclear Editor* to check cross section data.

MCNP 6.2 Editor

Configures the options for MCNP 6.2 input editing.



- *Turn tabs into spaces:* Turn tabulators into spaces during input. Avoids trouble with unrecognized characters (recommended).
- *Adjust tab width (first 5, then 2):* If a tab is turned into spaces, the first tab at the beginning of the line will be turned into 5 spaces (according to continuation lines), all other tabs will be turned into 2 spaces (recommended).
- *Show vertical line in column 5:* Shows a vertical line at the beginning of continuation lines.
- *End of line wrap:* Displays a vertical line at the maximum line length allowed (80 for MCNP up to 6.1.1, 128 for MCNP 6.2)

SERPENT 2

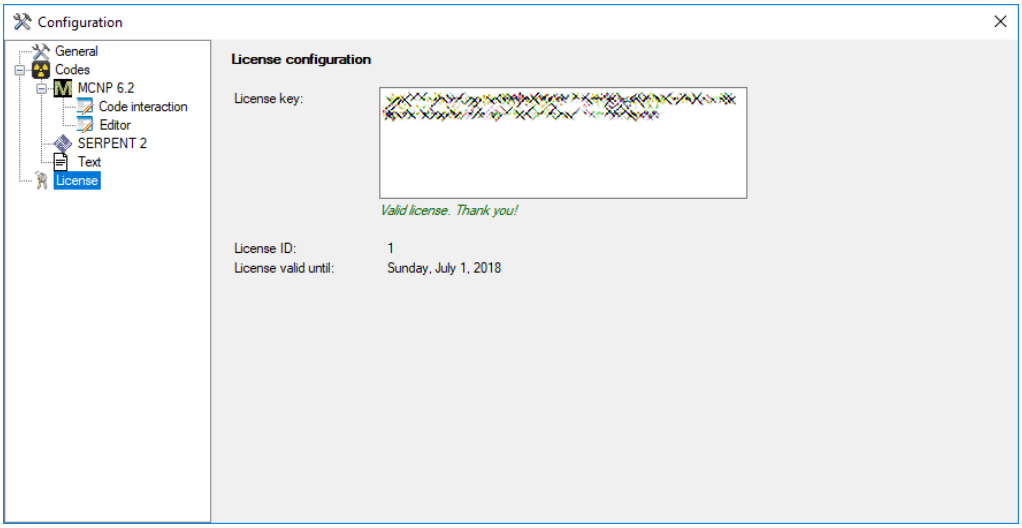
Nothing to configure yet.

TEXT

Nothing to configure yet.

License

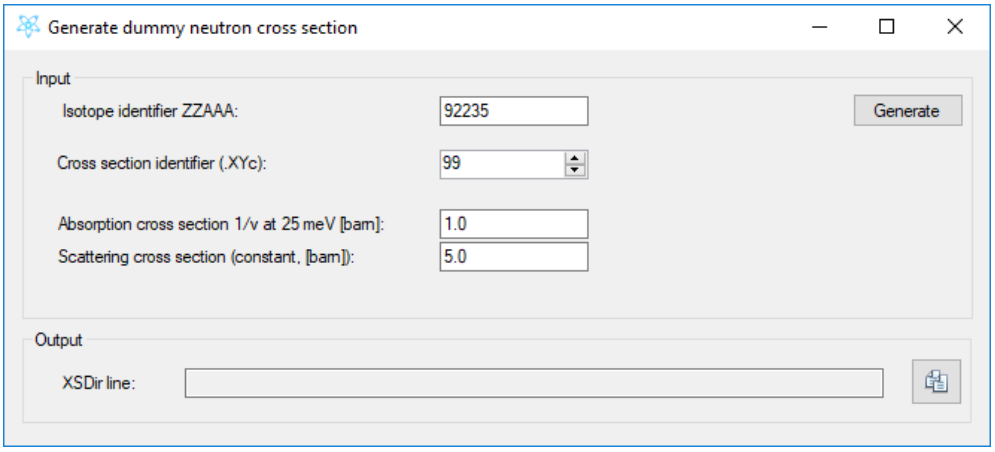
Enter the license key for the *Nuclear Editor* that you obtained when you purchased the software. Only with a valid license key, the full functionality of the *Nuclear Editor* will be unleashed. Trial keys are available for free on request.



Cross section tools

Dummy 1/v cross section

This tool allows you to generate a dummy cross section that only follows the 1/v-law. That might come in handy if you need to somehow "invent" cross sections for isotopes. Such cross sections are sometimes used in burn-up calculations for extremely rare and unimportant isotopes where the cross section has not yet been measured.



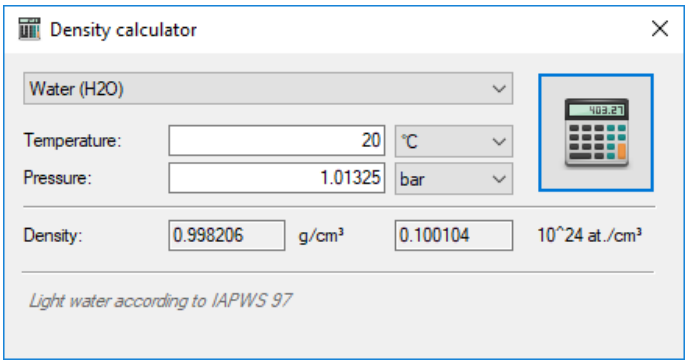
Naturally, be sure to know what you're doing when you're messing around with dummy cross sections.

Density calculation

The density calculation tool allows you to quickly calculate the density of the most common liquids and gases. Currently supported are:

- H2O (IAPWS 97)
- D2O (100%, 0-90 °C)
- Dry air near sea level (correct for humidity by using virtual temperature, $T = T_0(1 + 0.608\, q)$, where q is the relative humidity)

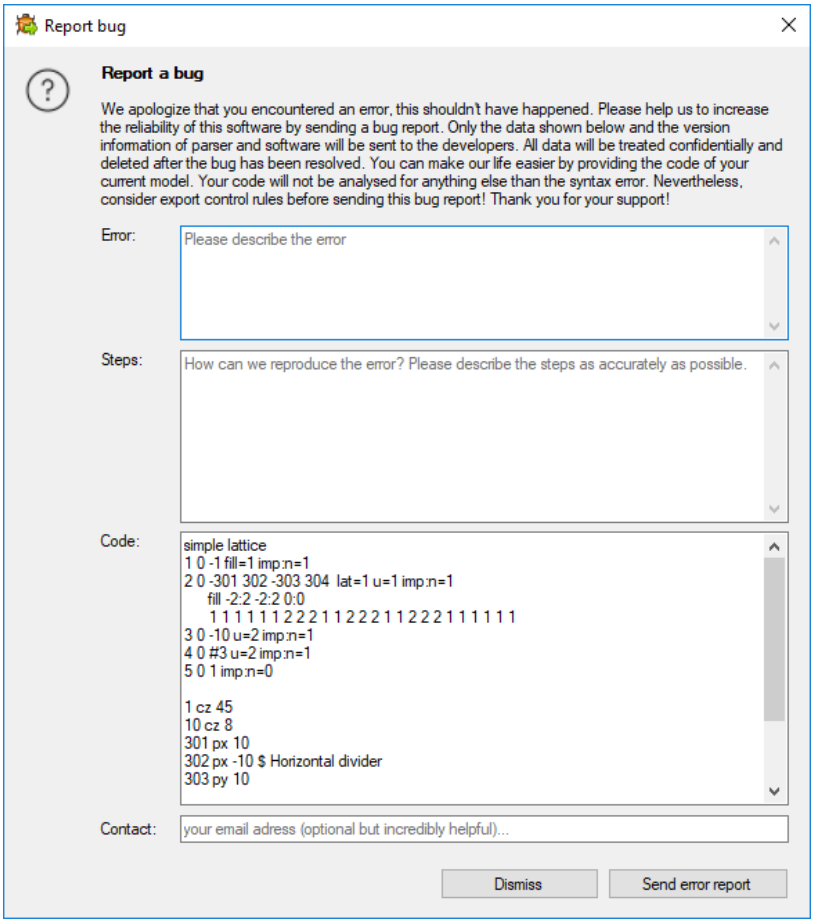
- He (293 - ~1800K, 1-100 bar)



Reporting bugs

If you run into unexpected trouble with the *Nuclear Editor*, you can report the bug using the `Help → Report Bug` menu. A dialog will appear that allows you to describe the error and the steps to reproduce it. Your currently opened model will also automatically appear. Remove this if export control or corporate rules do not allow you to share the model with the developers of the *Nuclear Editor*! In case of doubt, contact your legal department prior to sending the report. Please also include your email address so that we can contact you if we have further questions.

The same dialog also appears if one of the parsers of the *Nuclear Editor* crashes or runs into an unhandled exception.



SYSTEM REQUIREMENTS

The following is needed to properly run the *Nuclear Editor*:

- A decent PC (recommended: Core i5 or better, 8 Gb RAM, SSD)
- A decent monitor (recommended: 1920 x 1080 pixels or higher, 2nd monitor for additional tool windows)
- Windows 7 or better
- .NET 4.5 or better

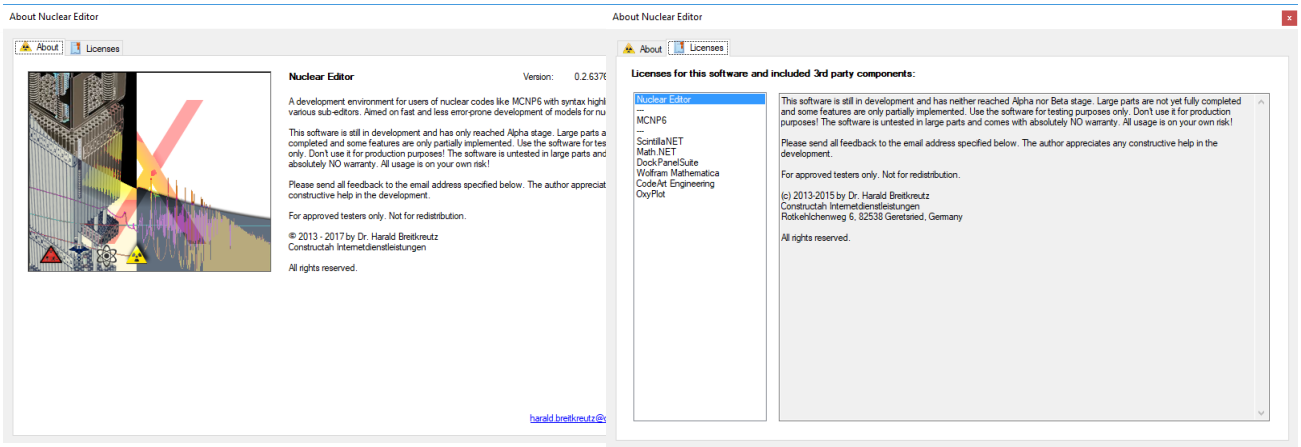
Optional:

- MCNP 6.2 or better for in-line running and for external plotting of MCNP input files[6]
- Running Serpent 2 is not yet supported
- Wolfram Mathematica 9 or better for 3D surface plotting in the surface editor for MCNP

COPYRIGHT AND 3RD PARTY LICENSES

To show licensing information of the *Nuclear Editor*, its sub-components and 3rd party libraries utilized by this software, go to `Help → About`.

[6] MCNPX 2.7 and MCNP 5 seem to work, too, but the *Nuclear Editor* has been designed for MCNP 6. Also, remember that the syntax parser checks for correct MCNP 6.2 syntax.



FAQ

Does the Nuclear Editor run on any other OS than Windows?
Currently: No. However, we are looking into making it available on Linux. It is not very high on the priority list, though[7]. Let us first fix all the bugs in the Windows UI.

My Firewall complains about outgoing connections from the Nuclear Editor!
The editor periodically checks for updates and uses online-information in the welcome window. It will transmit your current version and the license ID to the website. You can turn off update checking in the Options dialog.

Furthermore, if the *Nuclear Editor* runs into an exception that it cannot handle, it opens a dialog that offers you to send the current model and the stack trace to the developers, so that we can fix the error. If you choose to include your model (which we would greatly appreciate), please make sure that sending this model complies with export control regulations. The developers are in Germany, i.e. your model will be sent there.

My anti-virus software reports a Trojan hidden in the software!
If you obtained your copy through the official channels, there ought to be no Trojan in the software. However, some Antivirus programs identify the included licensing library as a Trojan because it uses some encryption techniques that seemingly are being used by Trojans, too.

Is it stable? Can I use it for production work?
Our recommendation is definitely not to use the editor for production work now. It is in alpha stage, crashes are likely and you will experience data loss. However, we all know you will use it anyway ☺. So, please save your work frequently and don't blame us - you have been warned!

Do I need a license for MCNP or Serpent to use the Nuclear Editor?
No. The *Nuclear Editor* does not contain code from any of the codes that it supports, i.e. it is independent. The parsers in the *Nuclear Editor* are written from scratch and not even inspired

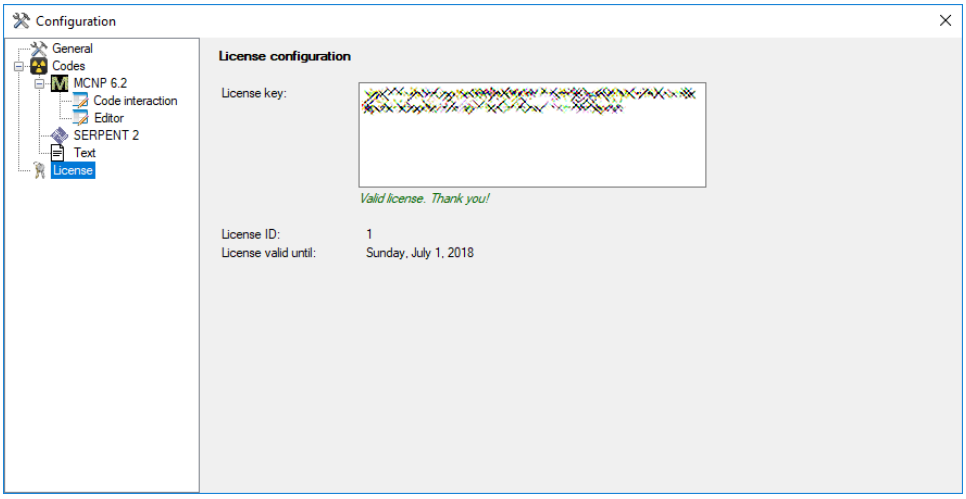
[7] The *Nuclear Editor* is written in C#, .NET 4.5. Basically, all the code is Mono-compatible and therefore could be run under Linux. Unfortunately, some of the libraries used by the *Nuclear Editor* contain pre-compiled binary code which is currently not compatible with Mono/Linux.

by the actual parsers of the codes the *Nuclear Editor* supports. This way, you can develop models for these codes without having an actual license for them. However, you will not be able to test and evaluate your models or use features of the *Nuclear Editor* that need to execute these programs.

Is this free software?

No. Do not give away the software to others! However, currently the *Nuclear Editor* is still in early development and therefore available free of charge to invited testers. If you would like to participate in the test, send a mail to beta@nuclearediter.com together with a short statement why you would be a good choice for testing.

Nevertheless, the *Nuclear Editor* already contains a licensing system. During the testing phase, a fixed license key is built into the software. This key expires every few month, giving you an additional incentive to update the software regularly. The license key can be viewed and exchanged in the `Tools → Options` menu and selecting `License` in the configuration dialog.



Once the *Nuclear Editor* is considered sufficiently stable, licenses will be available at a very reasonable price, with great discounts for students, code developers and active testers. The developers have put a huge amount of work into this software, several years, therefore we appreciate a little consideration for all the work we have done in our spare time. Yes, we have done that in our leisure time – we don't make a living from it, neither are we supported by a company. Thank you for your understanding ☺

APPENDIX

LIST OF SHORTCUTS

[F1]	Open manual
[⇧]+[F1]	Open code manual
[F5]	Execute code (MCNP6 only)
[⇧]+[F5]	Initialize code (MCNP6 only)
[F6]	Plot geometry with external plotter (MCNP6 only)
[⇧]+[F6]	Plot cross section with external plotter (MCNP6 only)
[Alt]+[→]	Go to next occurrence of object
[Alt]+[←]	Go to previous occurrence of object
[F2]	Rename object
[F7]	Open object editor (if available)
[F12]	Go to definition of object
[Ctrl]+[Alt]+[S]	Insert surface (MCNP6 only)
[Ctrl]+[Alt]+[M]	Insert material (MCNP6 only)
[Ctrl]+[Alt]+[T]	Insert transformation (MCNP6 only)
[Ctrl]+[Space]	Show autocomplete menu (MCNP6 only)
[Ctrl]+[F]	Find
[Ctrl]+[R]	Replace
[Ctrl]+[C]	Copy
[Ctrl]+[X]	Cut
[Ctrl]+[V]	Paste
[Ctrl]+[Z]	Undo
[Ctrl]+[Y]	Redo
[Ctrl]+[A]	Select all
[Ctrl]+[N]	New document
[Ctrl]+[O]	Open document
[Ctrl]+[P]	Print document
[Ctrl]+[S]	Save document
[Ctrl]+[F4]	Close document
[Alt]+[F4]	Close application