Nuclear Engineering Laboratory
M O N T E C U C C O L I N O
technical report
Nuclear Reactor Physics
Department of Energy and Nuclear Engineering
and of Environmental Control (DIENCA)
University of Bologna

Doc. ID: LIN-R08.1.2007
Subject: imcnp.vim
Date: December 3, 2007

Author: Giacomo Grasso
Phone: +39-051-6441708
e-mail: giacomo.grasso@mail.ing.unibo.it

A Vim syntax highlighter for MCNP input files

Whatever MCNP simulation is completely defined by means of the geometrical, material, operative and numerical positions implemented in the proper input file. As soon as the simulating systems become more and more complex, the input file structure and length grow heavier, making the file itself little readable; such simulations also require great computational powers, which are usually not accessible via graphical interfaces: this implies that it is often required to modify the MCNP input file by means of shell editors like Vim.
To simplify the readability of such files it has therefore been developed a proper syntax highlighter for the Vim editor, supporting all the main functions useful for easy access and browsing of MCNP input files.

The Author
Giacomo Grasso

The Supervisor
prof. Marco Sumini

Revisions List:

Rev. No	Date	Author
1	*December 18, 2007*	*Giacomo Grasso*

A Vim syntax highlighter for MCNP input files

Giacomo Grasso

December 3, 2007

Abstract

Whatever MCNP simulation is completely defined by means of the geometrical, material, operative and numerical positions implemented in the proper input file. As soon as the simulating systems become more and more complex, the input file structure and length grow heavier, making the file itself little readable; such simulations also require great computational powers, which are usually not accessible via graphical interfaces: this implies that it is often required to modify the MCNP input file by means of shell editors like Vim.
To simplify the readability of such files it has therefore been developed a proper syntax highlighter for the Vim editor, supporting all the main functions useful for easy access and browsing of MCNP input files.

Contents

1 Introduction

The readability of MCNP [1] input files is fundamental in order to keep easy the development of complex systems descriptions for simulation. Although a tidy organizing of the file structure and sectioning, together with a wide and clear commenting, may help at easily maintaining a huge input file, the possibility to highlight the MCNP syntax, separating keywords from comments and numbers, would simplify very much the identification of "active" sections from comments and further make the recognition of different sections easier.

Since the great diffusion of MCNP under *nix systems has both historical backgrounds and computational reasons, it has been developed a syntax highlighter descriptor for MCNP input files to use within the Vim [2] shell editor. It is indeed often necessar to open and edit input files from remote shells on simulation workstation or clusters, and it is often more convenient to use shell editors for quick bringing changes to the simulation locally on the host machine. Besides the Vim editor, also available for Windows systems, is a preferential choice for many developers used to access *nix machines via command line.

2 Description of the syntax highlighter

The Vim syntax highlighter has been developed through the definition of a new fyletype, i.e. the *imcnp* - MCNP input file. This let the editor associate the proper syntax definition to opening MCNP input files, and therefore to highlight specified keywords with the assumed color scheme.

In the developed version 0.1 of the `imcnp.vim` syntax descriptor file (presented in appendix A) the most common keywords have been included, in order to cover the largest part of common cases. Keywords have been moreover classified into different sets, each one with a proper highlighting rule. The main categories implemented in the present version of the descriptor are:

Header the first line of the input file, whether this is not a comment card, used in MCNP as a descriptive label for output sectioning;

Keyword all the common commands for cell description, source specification and simulation parameters definition, together with the most common operators (i.e. “=”, “*”, “#”, “<”);

Type all the common typologies of predefined data, such as surfaces or macrobodies, source distributions and cross sections labels;

Todo all the materials and tallies cards;

Number all integer number lables referring to cells, surfaces, materials, tallies and simulation parameters;

Float all floating numbers intended as simulation variables values;

Comment all comment lines and in line notes.

The color scheme for such categories has been kept from Vim default, to carry on the visual association.

3 Installation and use instructions

The procedure for installation and use of the `imcnp.vim` syntax descriptor file are here presented: the few steps required only need to prior copying the text of the descriptor presented in appendix A in a new file named `imcnp.vim`.

3.1 Installation

The installation of the `imcnp` syntax descriptor file might follow two different procedures, either as administrator or as normal user. The only difference between the two is that the former can lead to a common installation for all system users, while the latter is only owned by the single user.

3.1.1 Administrator's procedure

To install the `imcnp.vim` syntax descriptor file as root (that is for all system users) it is needed to prior verify if the syntax highlighting is already turned on: it can be done either by opening a common language source file (for which it is supposed to be already available the syntax descriptor within Vim) or looking at the `vimrc` configuration file in the Vim installation directory (for GNU/Linux systems, it is usually found under `/usr/share/vim/`) for the line

```
syntax on
```

and check whether it is preceded by the comment mark `"` which, in case, must be removed.

After that, two easy steps are required to complete the procedure. The first is to copy the `imcnp.vim` file into the `/usr/share/vim/vimxx/syntax/`[1] folder. At last the following lines must be added to the `/usr/share/vim/vimxx/filetype.vim` file to define a new filetype to be associated to the installing *imcnp* syntax highlighting descriptor:

```
" MCNP input
au BufNewFile,BufRead *.imcnp                    setf imcnp
```

3.1.2 Single user's procedure

The installation of the `imcnp.vim` syntax descriptor file as user is quite similar to the administrator one but in the location of the Vim configuration folder (due to the limited access rights to system folders). As in the previous case, it is needed to verify if the syntax highlighting is already turned on: the easiest way is again by opening a common language source file (for which it is supposed to be already available the syntax descriptor within Vim). If the syntax highlighting is disabled, or in case of doubt, it can be turned on by the two alternative commands

```
:syntax enable
```

[1] *xx* indicates the Vim version number (version and subversion only, without dot): it can be retrieved by typing *vim -v*. For example, the *xx* for *version 7.0.235* becomes *70*.

and

```
:syntax on
```

The difference between the two is that the former enables the user defined syntax highlighting scheme eventually defined, while the latter overrides any highlighting personal setting by loading the default options.

After that, the user has to check for a `.vim` personal Vim configuration folder to exist in his home, and to create it whether it does not. Once the configuration directory exists, the user has to make a `syntax` subfolder into the latter and, finally, to copy the `imcnp.vim` syntax descriptor file within it.

3.2 Use

Once the *imcnp* syntax descriptor has been correctly installed, the Vim user can open any MCNP input file and enable the highlighting for it just typing

```
:set syntax=imcnp
```

Only in the case that the syntax descriptor file has been installed as root, the user could also temporally rename the MCNP input file by adding the fictitious extension `.imcnp` to the filename to let Vim automatically recognize the filetype and enable the proper syntax highlighting.

A imcnp.vim

```
" Vim syntax file
" Language: MCNP input file
" Version: 1.01
" Last Change: 2007 Dec. 19
" Maintainer: Giacomo L.A. Grasso
" Nuclear Engineering Laboratory - LIN - of Montecuccolino
" Dept. of Energy and Nuclear Engineering and of Environmental
" Control (DIENCA) - University of Bologna
" Contact: <giacomo.grasso@mail.ing.unibo.it>
" Usage: Do :help imcnp-syntax from Vim
" Credits:
"  Version 0.1 was based on the MCNP5 input file scheme as reported in the
"  Los Alamos User Manual. For instructions on use, do :help imcnp from vim

" For version 5.x: Clear all syntax items
" For version 6.x: Quit if a syntax file is already loaded
if version < 600
  syntax clear
elseif exists("b:current_syntax")
  finish
endif

syn case ignore

" ------------- Generals ----------------------------------------------------
" --- Unprocessed
" Header
syn match imcnpUnitHeader display "\%1l.\+$"
" Comments
syn match imcnpComment display "\%1cc$"
syn match imcnpComment display "\%1cc\s.*$"
syn match imcnpComment display "\$.*$"
" --- Commons
" Characterizig
syn keyword imcnpKeyword n
" --- Numbers of various sorts
" Integers
syn match imcnpLabelNumber display "[-+*#]\=\d\+r\="
" floating points
syn match imcnpFloat display "[-+]\=\d\{2,}\.\d*"
syn match imcnpFloat display "[-+]\=\d\=\.\d*\(e[-+]\=\d\+\)\="

" ------------- Cell cards --------------------------------------------------
" descriptives
syn keyword imcnpKeyword like but imp ext mat rho vol tmp area trcl u fill lat

" ------------- Surface cards -----------------------------------------------
" Surfaces
syn keyword imcnpType px py pz p cx cy cz so s sx sy sz kx ky kz sq gq tx ty tz x y z
syn match imcnpType "\<[ck]/[xyz]\>"
" Macrobodies
syn keyword imcnpType box rpp sph rcc rhp hex rec trc ell wed arb
```

```
" ------------- Tally cards ----------------------------------------------------
" --- Materials cards
" mat number
syn match imcnpTodo "m[t]\=\d\+"
" ENDF library
syn match imcnpType display "\<\d\{2,6}\.\d\d[cdytpuemg]\>"
" --- Tallies cards
" Tally
syn match imcnpTodo "f\(\(mesh\)\|[cqmsut]\)\=\d*"
syn match imcnpTodo "e[m]\=\d*"
syn match imcnpTodo "t[mf]\=\d\+"
syn match imcnpTodo "c[mf]\=\d\+"
syn match imcnpTodo "d[efd\(xt\)]\=\d\+"
syn match imcnpTodo "s[fd]\=\d\+"
syn keyword imcnpKeyword geom origin imesh iints jmesh jints kmesh kints emesh eints out factor
" Comments
syn region imcnpComment matchgroup=imcnpTodo start="[sf]c\d\+" end="$" contains=imcnpComment
" syn match imcnpComment
" --- Source cards
" Generic
syn keyword imcnpKeyword ksrc sdef
" Definition
syn keyword imcnpKeyword ssw ssr ipt icl jsu cel sur par tr pos rad axs dir vec nrm ccc ara eff erg
        tme wgt
" Distribution
"syn match imcnpTodo "d\d\+"
"syn match imcnpTodo "s[ipbc]\d\+"
"syn match imcnpTodo "ds\d\+"
"syn match imcnpTodo "tr\d\+"
" --- Mode cards
" Simulation
syn keyword imcnpKeyword mode kcode
" Output
syn keyword imcnpKeyword print
syn keyword imcnpType col cf ij ik jk

"Catch errors caused by too many right parentheses
syn region imcnpParen transparent start="(" end=")" contains=ALLBUT,imcnpParenError,@imcnpCommentGroup,
        cIncluded,@spell
syn match  imcnpParenError   ")"
syn region imcnpParen transparent start="\[" end="\]" contains=ALLBUT,imcnpParenError,@imcnpCommentGroup,
        cIncluded,@spell
syn match  imcnpParenError   "\]"

syn match imcnpOperator "="
syn match imcnpOperator ":"
syn match imcnpOperator "<"

" Define the default highlighting.
" For version 5.7 and earlier: only when not done already
" For version 5.8 and later: only when an item doesn't have highlighting yet
if version >= 508 || !exists("did_imcnp_syn_inits")
  if version < 508
    let did_imcnp_syn_inits = 1
    command -nargs=+ HiLink hi link <args>
```

```
  else
    command -nargs=+ HiLink hi def link <args>
  endif

  HiLink imcnpKeyword   Keyword
  HiLink imcnpConstructName Identifier
  HiLink imcnpConditional Conditional
  HiLink imcnpRepeat Repeat
  HiLink imcnpTodo Todo
  HiLink imcnpContinueMark Todo
  HiLink imcnpString String
  HiLink imcnpNumber Number
  HiLink imcnpOperator Operator
  HiLink imcnpBoolean Boolean
  HiLink imcnpLabelError Error
  HiLink imcnpObsolete Todo
  HiLink imcnpType Type
  HiLink imcnpStructure Type
  HiLink imcnpStorageClass StorageClass
  HiLink imcnpUnitHeader Identifier
  HiLink imcnpReadWrite Keyword
  HiLink imcnpIO Keyword
  HiLink imcnp90Intrinsic Function

  HiLink imcnpInclude Include

  HiLink imcnpLabelNumber Special
  HiLink imcnpTarget Special
  HiLink imcnpFormatSpec Identifier

  HiLink imcnpFloat Float
  HiLink imcnpPreCondit PreCondit
  HiLink imcnpInclude Include
  HiLink cIncluded imcnpString
  HiLink cInclude Include
  HiLink cPreProc PreProc
  HiLink cPreCondit PreCondit
  HiLink imcnpParenError Error
  HiLink imcnpComment Comment
  HiLink imcnpSerialNumber Todo
  HiLink imcnpTab Error

  delcommand HiLink
endif

let b:current_syntax = "imcnp"

" vim: ts=8 tw=132
```

B imcnp.xml for katepart

The same vim syntax scheme has been rewritten to suite with the kate highlighting engine (katepart [3]), in order to be supported in every visual editor of the K Desktop Environment (KDE).

The following source must be copied in a `imcnp.xml` file within either the

```
$(PREFIX)/share/apps/katepart/syntax/
```

folder (for whole system installation) or the

```
~/.kde/share/apps/katepart/syntax/
```

one (for single user installation).

To use the MCNP input file syntax descriptor, it is possible to add a "text/x-imcnp" MIME type for `*.imcnp` files, to let the system automatically recognize MCNP input files with the proper, fictitious extension and load the given syntax highlighter. Otherways, the syntax can be highlighted manually by following the "Highlighting → Sources → imcnp" path in the "Tools" menu of every KDE editor.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE language SYSTEM "language.dtd">
<!--
Copyright (C) 2007 Giacomo L.A. Grasso <giacomo.grasso@mail.ing.unibo.it>.
Nuclear Reactor Physics group
Nuclear Engineering Laboratory (LIN) of Montecuccolino
Department of Energy and Nuclear Engineering and of Environmental Control (DIENCA)
University of Bologna - ITALY
-->
<language name="imcnp" version="1.01" kateversion="2.5" section="Sources" extensions="*.imcnp"
mimetype="text/x-imcnp" casesensitive="0" author="Giacomo L.A. Grasso
(giacomo.grasso@mail.ing.unibo.it)" license="LGPL">
  <highlighting>

    <list name="keywords">
      <item> like </item>
      <item> but </item>
      <item> imp </item>
      <item> ext </item>
      <item> mat </item>
      <item> rho </item>
      <item> vol </item>
      <item> tmp </item>
      <item> area </item>
      <item> trcl </item>
      <item> u </item>
      <item> fill </item>
      <item> lat </item>
      <item> ssw </item>
```

```
      <item> ssr </item>
      <item> ipt </item>
      <item> icl </item>
      <item> jsu </item>
      <item> cel </item>
      <item> sur </item>
      <item> par </item>
      <item> tr </item>
      <item> pos </item>
      <item> rad </item>
      <item> axs </item>
      <item> dir </item>
      <item> vec </item>
      <item> geom </item>
      <item> origin </item>
      <item> imesh </item>
      <item> iints </item>
      <item> jmesh </item>
      <item> jints </item>
      <item> kmesh </item>
      <item> kints </item>
      <item> emesh </item>
      <item> eints </item>
      <item> out </item>
      <item> factor </item>
    </list>
    <list name="modular">
      <item> mode </item>
      <item> kcode </item>
      <item> sdef </item>
      <item> ksrc </item>
      <item> print </item>
    </list>
    <list name="modes">
      <item> n </item>
<!--      <item> e </item>
      <item> p </item>   -->
    </list>
    <list name="geotypes">
      <item> px </item>
      <item> py </item>
      <item> pz </item>
      <item> p </item>
      <item> cx </item>
      <item> cy </item>
      <item> cz </item>
      <item> so </item>
      <item> s </item>
      <item> sx </item>
      <item> sy </item>
      <item> sz </item>
      <item> kx </item>
      <item> ky </item>
      <item> kz </item>
      <item> sq </item>
```

```
    <item> gq </item>
    <item> tx </item>
    <item> ty </item>
    <item> box </item>
    <item> rpp </item>
    <item> sph </item>
    <item> rcc </item>
    <item> rhp </item>
    <item> hex </item>
    <item> rec </item>
    <item> trc </item>
    <item> ell </item>
    <item> wed </item>
    <item> arb </item>
  </list>
  <list name="outtypes">
    <item> col </item>
    <item> cf </item>
    <item> ij </item>
    <item> ik </item>
    <item> jk </item>
  </list>

  <contexts>
    <context attribute="Normal Text" lineEndContext="#stay" name="default" >
      <IncludeRules context="find_comments" />
      <IncludeRules context="find_todos" />
      <IncludeRules context="find_types" />
      <IncludeRules context="find_numbers" />
      <IncludeRules context="find_symbols" />
      <IncludeRules context="find_header" />
      <IncludeRules context="find_intrinsics" />
    </context>
    <!--************************END OF THE MAIN CONTEXT*************************-->

    <!-- This context highlights preprocessor lines -->
    <context attribute="Normal Text" lineEndContext="#stay" name="find_header">
      <RegExpr attribute="Elemental Procedure" context="#stay" String="(cDEC\$|CDEC\$).*$"
               row="0"/>
    </context>

    <!-- This context highlights comments -->
    <context attribute="Normal Text" lineEndContext="#stay" name="find_comments">
      <RegExpr attribute="Comment" context="#stay" String="[cC](\s.+)?$" column="0"/>
      <RegExpr attribute="Comment" context="#stay" String="\$.*$"/>
    </context>

    <context attribute="" lineEndContext="#stay" name="find_comments2">
      <RegExpr attribute="Inquiry Function" context="find_comments2" String="[sf]c\d+"/>
      <RegExpr attribute="Comment" context="#pop" String=".*$"/>
    </context>

    <!-- This context highlights symbols -->
    <context attribute="Normal Text" lineEndContext="#stay" name="find_symbols">
      <AnyChar attribute="Keyword" context="#stay" String="=:&lt;"/>
```

```
      <AnyChar attribute="Symbol" context="#stay" String="()"/>
      <AnyChar attribute="Symbol" context="#stay" String="[]"/>
    </context>

    <!-- The following two contexts match declarations -->
    <context attribute="Normal Text" lineEndContext="#stay" name="find_todos">
      <RegExpr attribute="Inquiry Function" context="#stay" String="m[t]?\d+" insensitive="TRUE"/>
      <RegExpr attribute="Inquiry Function" context="#stay" String="d[efd(xt)]?\d+"
              insensitive="TRUE"/>
      <RegExpr attribute="Inquiry Function" context="#stay" String="s[ifbd]?\d+"
              insensitive="TRUE"/>
      <RegExpr attribute="Inquiry Function" context="#stay" String="ds\d+" insensitive="TRUE"/>
      <RegExpr attribute="Inquiry Function" context="#stay" String="tr\d+" insensitive="TRUE"/>
      <RegExpr attribute="Inquiry Function" context="#stay" String="\bf([qmsut]|mesh)?\d*\b"
              insensitive="TRUE"/>
      <RegExpr attribute="Inquiry Function" context="#stay" String="\be[m]?\d*\b"
              insensitive="TRUE"/>
      <RegExpr attribute="Inquiry Function" context="#stay" String="t[mf]?\d+" insensitive="TRUE"/>
      <RegExpr attribute="Inquiry Function" context="#stay" String="c[mf]?\d+" insensitive="TRUE"/>
      <RegExpr attribute="Inquiry Function" context="find_comments2" String="[sf]c\d+"/>
    </context>

    <context attribute="Normal Text" lineEndContext="#stay" name="find_types">
         <RegExpr attribute="Data Type" context="#stay" String="[ck]/[xyz]"/>
         <RegExpr attribute="Data Type" context="#stay" String="\b[0-9]+\.[0-9]+[cdytpuemg]{1}\b"/>
    </context>

    <!-- The following context matches intrinsic procedures -->
    <context attribute="Normal Text" lineEndContext="#stay" name="find_intrinsics">
      <keyword attribute="Keyword" context="#stay" String="keywords"/>
      <keyword attribute="Data Type" context="#stay" String="geotypes"/>
      <keyword attribute="Data Type" context="#stay" String="outtypes"/>
      <keyword attribute="Preprocessor" context="#stay" String="modes"/>
      <keyword attribute="IO Function" context="#stay" String="modular"/>
    </context>

    <!-- The following context matches integer and real numbers -->
    <context attribute="Normal Text" lineEndContext="#stay" name="find_numbers">
      <!-- Floating-point numbers with optional kind -->
      <RegExpr attribute="Float" context="#stay" String="[0-9]*\.[0-9]+([e][+-]?[0-9]+)?"
              insensitive="TRUE"/>
      <RegExpr attribute="Float" context="#stay" String="\b[0-9]+\.[0-9]*([e][+-]?[0-9]+)?"
              insensitive="TRUE"/>
      <RegExpr attribute="Float" context="#stay" String="\b[0-9]+[e][+-]?[0-9]+"
              insensitive="TRUE"/>
      <!-- Integers with optional kind specifier -->
      <RegExpr attribute="Decimal" context="#stay" String="\b[\*-]?[0-9]+r?"/>
      <!-- Integers in binary, octal and hexadecimal notations -->
    </context>
  </contexts>
  <itemDatas>
    <itemData name="Normal Text" defStyleNum="dsNormal"/>
    <itemData name="Keyword" defStyleNum="dsKeyword"/>
    <itemData name="Data Type" defStyleNum="dsDataType"/>
    <itemData name="Decimal" defStyleNum="dsDecVal"/>
```

```
      <itemData name="Float" defStyleNum="dsFloat"/>
      <itemData name="Comment" defStyleNum="dsComment"/>
      <itemData name="Symbol" defStyleNum="dsNormal"/>
      <itemData name="Preprocessor" defStyleNum="dsOthers"/>
      <itemData name="Operator" defStyleNum="dsKeyword" color="#008000" selColor="#ff00ff" bold="1"
                italic="0"/>
      <itemData name="IO Function" defStyleNum="dsFunction" color="#006060" selColor="#ffffff"
                bold="0" italic="0"/>
      <itemData name="Elemental Procedure" defStyleNum="dsKeyword" color="#600060" selColor="#ffa0ff"
                bold="1" italic="0"/>
      <itemData name="Inquiry Function" defStyleNum="dsFunction" color="#000060" selColor="#a0a0ff"
                bold="1" italic="1"/>
    </itemDatas>
  </highlighting>
  <general>
    <comments>
      <comment name="singleLine" start="c"/>
      <comment name="singleLine" start="\$"/>
    </comments>
    <keywords casesensitive="0"/>
  </general>
</language>
<!--
    // kate: space-indent on; indent-width 5; replace-tabs on;
-->
```

References

[1] F. B. Brown *et al.* MCNP Version 5. *Trans. Am. Nucl. Soc.*, 87:273, 2002.

[2] The vim text editor. http://www.vim.org/.

[3] Katepart: a fast and featurerich text editor component. http://kate-editor.org/katepart.