

SEP #12 : Scipad / Add support for different encodings

Title: Scipad / Add support for different encodings

Version: 2.1

Author: Francois Vogel – fvogelnew1@free.fr

Review:

Commented:

State: Accepted (consensus reached on dev. list)

Scilab-Version: Master – This new feature is included in Scipad 7.14

Vote: No

Created: 15/12/08

Abstract

This SEP proposes to enhance Scipad by adding support for other encodings than the system encoding. The user will then be able to edit files encoded in UTF-8 or any other available encoding.

Rationale

Scilab development is currently heading toward more internationalization. This trend brings the need for editing files containing text encoded in possibly different encoding than the system encoding.

This is especially true when dealing with Scilab XML help files, which are a frequent use case for developers.

The proposed enhancement aims at providing users the ability to load, edit and save text files in the encoding of their choice (that is: encodings the underlying Tcl installation knows about), all from within Scipad.

Auto-detection of encoding of XML files is also provided.

Implementation

Implementation is based on the Tcl command *encoding*, see its documentation here:

<http://www.tcl.tk/man/tcl8.5/TclCmd/encoding.htm>

The default system encoding is the platform default encoding that will be detected at Scipad launch.

A new "Encoding" menu will be added to the existing Options menu of Scipad. It will be architected in two parts: a MRU (Most Recently Used) list of encodings at the top, and less frequently used entries at the bottom.

The list of all the available encodings (i.e. *[encoding names]*) will be shown in a series of radiobuttons gathered in a sub-menu labeled "More encodings". Available encodings will be numerous, which is the reason why the MRU list of encodings is created for quick access.

The initial list of MRU encodings will include the platform default encoding, iso8859-1 (used by the Scilab source files (File/Open source of...)), and utf-8. The user will be able to select the size of the MRU list between zero and 10 elements. Whenever the user selects an encoding from the "More encodings" list, this encoding will be added in the MRU list of encodings (if not already there).

The encoding name attached to a file will be a new property of each Scipad buffer: *\$listoffile("\$textarea",encoding)*. It is therefore not a global setting but is local and may be different for each buffer.

When the user selects an item from the Encoding menu, it sets *\$listoffile("\$textarea",encoding)* to the selected encoding name.

When a file is read from disk or saved, *\$listoffile("\$textarea",encoding)* is used to configure the channel (*fconfigure \$ch -encoding ...*), ensuring proper encoding of the file is exported to the OS (when saving) or used for file content interpretation (when loading).

The encoding used by Tcl to talk to the operating system is never changed though (i.e. *encoding system* is never set). It must be so because The system encoding of the platform does never change in the process, only the encoding used to read and save files (that is: their content, not their names) may change.

Whenever the encoding currently used is different from the platform default encoding, this will be shown to the user in the Scipad title bar (and tile tile, if applicable) by appending a tag such as "(utf-8)". This is much the same principle as for the already existing "(modified)" or "(readonly)" tag. When the file encoding matches the default encoding, no tag will be appended, thus reducing unnecessary clutter.

No additional tag will be added to the file names stored in the Windows menu of Scipad.

The Scipad preferences file must be saved in the default platform system encoding, in order to preserve filepaths for later use from the MRU list in the File menu. Otherwise an attempt to open a recent file having special characters in its pathname may fail if the encoding currently selected in Scipad (which is used by the OS to parse the filepath) is not the platform system encoding.

More generally, and for similar reasons, most files used internally by Scipad will always be read and saved using the platform default system encoding. This includes:

- the Scipad preferences file (read and written by Scipad)
- the temporary file that ScilabEval_It will create if the command length is longer than bsiz (written by Scipad, read and executed by Scilab)
- the temporary file the debugger uses for exec-ing the non level zero code just before running the configured function (written by Scipad, read and executed by Scilab)
- the BUGS file (read by Scipad)
- the changelog.txt file (read by Scipad)
- the AddingTranslations.txt file (read by Scipad)
- the log file that can be used for debugging Scipad (written by Scipad, normally not geared in)

Other internal files will use the current encoding:

- the temporary file created for printing an unsaved buffer (both on Windows and on Linux) (written by Scipad, read and executed by the underlying OS)

When opening source files from Scipad through the File/Open source of ... command, the encoding will be forced to iso8859-1. Such source files include library function (macros) files, Scicos (macros) source files, and Scicos blocks source files.

This SEP also covers auto-detection of encodings upon file reading. This is limited to XML files¹,

¹ More information about the general process of identifying the encoding of an arbitrary (including XML) text file:

<http://chardet.feedparser.org/docs/faq.html>

<http://feedparser.org/docs/character-encoding.html>

[http://msdn.microsoft.com/en-us/library/aa741220\(VS.85\).aspx](http://msdn.microsoft.com/en-us/library/aa741220(VS.85).aspx)

which is typical in the Scilab workflow when editing help files in different locales. The XML prolog will be parsed for the "encoding" attribute. In case such parsing fails, or if the encoding name found does not match any known encoding, Scipad will silently not attempt to change the encoding currently selected by the user. In case the parsing succeeds, the XML file will be read again with the matched encoding name. How to regexp match the encoding declaration in the XML prolog is precisely described in the XML specification <http://www.w3.org/TR/xml>. This specification will be followed as closely as it is possible within a single regexp in Tcl. Writing (or integrating) a full fledged XML parser is out of the scope of this SEP.

Auto-detection of encoding in XML files will be a new saved preference located at the bottom of the Options/Encodings menu.

The whole implementation of this SEP will only impact the scipad module of Scilab.

Typical use cases in Scipad

If the file is an XML file containing an encoding declaration

1. Open the file. Scipad opens it directly with the correct encoding matched from the XML prolog.
2. Fiddle with the content, add text, possibly switch buffers, select other encodings, etc.
3. Before saving, select the encoding that Scipad shall use for saving the file.
4. Save the file.

If the encoding of the file to be read IS known beforehand

1. Select the encoding with which the next file should be read.
2. Open the file.
3. Fiddle with the content, add text, possibly switch buffers, select other encodings, etc.
4. Before saving, select the encoding that Scipad shall use for saving the file.
5. Save the file.

If the encoding of the file to be read IS NOT known beforehand

1. Open the file.
 2. Damned, it's not the right encoding. Select the right encoding through the options menu. There is no visual effect so far (this is not encoding conversion).
 3. Do any modification to the file in order to ungrey the File/Revert command.
 4. File/Revert. The file is now displayed with the selected encoding. Note that it's a reload of the file from the disk: changes will be lost and the file will be read again, this time with the correct encoding.
 5. Fiddle with the content, add text, possibly switch buffers, etc.
 6. Optionally, before saving, select another encoding that Scipad shall use for saving the file.
 7. Save the file.
-

Alternatives

None that I can see at the moment.

It has been suggested to add an "Encoding" button in the file open/save dialogs themselves. On Windows this is just not possible because Tk is calling the platform native dialogs, which do not provide the ability to select an encoding. On Linux it would be possible, but it would mean tweaking the source code provided by the Tk package. This is not something we want to do, because it would cause a lot of headaches whenever the Tk version gets updated. Implementation of the file chooser is not part of the public interface of Tk. The public interface is limited to the single command `tk_getOpenFile`, which has no existing option to show an encoding pull down menu.

Compatibility

This feature will be added in Scilab master GIT branch. There is no specific compatibility issue foreseen for the newly provided features covered by this SEP.

Limitations

1. Encoding conversion

Encoding conversion is NOT part of this SEP.

This feature would take the current buffer as input. This content is supposed to be encoded in the currently selected encoding (encoding A). Scipad would then convert this content into a different encoding B, which would as a side effect become the current encoding for the file. This would be done in place in the current buffer.

This encoding conversion function would certainly be handy, for instance when a file has been opened with the wrong encoding. Encoding conversion could then be used to check visually that the new encoding better fits the file content.

This feature would be based on Tcl commands *encoding convertfrom* and *encoding convertto*.

The present SEP could be augmented to include an encoding conversion feature. However, the discussion about this SEP on the dev. mailing list turned out to prove there is no interest for it

Besides, preliminary implementation tries proved it is much less straightforward than expected.

This feature is therefore left aside from this SEP.

Detailed reference implementation

There is one new file MRUlist.tcl in the codebase, and a number of diffs.

The diffs provided below are against the master git branch dated 10 Jan. 2009 (Scipad 7.13).

The new file is provided below after the diffs. It shall be placed in SCI/modules/scipad/tcl

Diffs against the most recent Scipad codebase:

```
--- tcl/buffernavigation.tcl Tue Jan 06 00:45:32 2009
+++ tcl/buffernavigation.tcl Sat Jan 10 11:57:06 2009
@@ -23,6 +23,7 @@
#
# See the file scipad/license.txt
#
+
#####
#
# About the implementation of paned windows:
@@ -359,6 +360,7 @@
```

```

# Set all the settings such that $textarea becomes the current one
global pad $listoffile($textarea,language)
global ColorizeIt $listoffile($textarea,colorize)
global buffermodifiedsincesearch
+ global currentencoding

# clear the selection when leaving a buffer - check first that the
# textarea still exists because it might have been destroyed when
@@ -377,6 +379,7 @@
keyposn $textarea
set Scheme $listoffile("$textarea",language)
set ColorizeIt $listoffile("$textarea",colorize)
+ set currentencoding $listoffile("$textarea",encoding)
schememenu $textarea
highlighttextarea $textarea
TextStyles $textarea
@@ -547,6 +550,7 @@
# it creates a new empty textarea
global winopened $listoffile($newta,language)
global listoftextarea pad
+ global defaultencoding

# ensure that the cursor is changed to the default cursor
event generate [gettextarea] <Leave>
@@ -565,6 +569,7 @@
set $listoffile("$pad.new$winopened",undostackdepth) 0
set $listoffile("$pad.new$winopened",redostackdepth) 0
set $listoffile("$pad.new$winopened",progressbar_id) ""
+ set $listoffile("$pad.new$winopened",encoding) $defaultencoding
lappend listoftextarea $pad.new$winopened

addwindowmenuentry $winopened $listoffile("$pad.new$winopened",displayedname)
@@ -618,6 +623,7 @@
set $listoffile("$newta",undostackdepth) $listoffile("$ta",undostackdepth)
set $listoffile("$newta",redostackdepth) $listoffile("$ta",redostackdepth)
set $listoffile("$newta",progressbar_id) $listoffile("$ta",progressbar_id)
+ set $listoffile("$newta",encoding) $listoffile("$ta",encoding)
lappend listoftextarea $newta

addwindowmenuentry $winopened $listoffile("$pad.new$winopened",displayedname)
--- BUGS Tue Jan 06 21:34:22 2009
+++ BUGS Sun Jan 11 11:00:57 2009
@@ -38,34 +38,21 @@
-Some shortcuts may not work on certain platforms. E.g. Control-F1 does not on
at least one Mandrake 10.1 Official Linux. Was also found on a RH8 machine.

--When a function definition line ends with a comment, the line numbering is
- wrong 1 unit, resulting in various strange behaviors: execution not stopping
- at breakpoints or not up-to-date value of variables in the watch window
- This is Scilab bug 1469 (outside of Scipad) that deals with the general
- issue of mismatch between line numbering as reported by fun2string and that
- held by the breakpoint handling
-
-logical line numbering issue:
function a=foo(); a=1;
a=2;
endfunction
In scipad, a=1 is on line 1; for scilab it is on line 2 of function foo
- Also related to bug 1469 - better wait for clarification on this one instead
+ This is bug 3911 - better wait for clarification on this one instead
of working around Scilab parser bugs

--step-by-step debugging may highlight the wrong line in Scipad when continued
- lines without dots are present before the breakpoint stop. Bug 1469 again!!!
-
--Bug 1469 once more in the following case (mind the spaces before
- "endfunction"):
+-Bug 3911 also in the following case
function test
- a=1
- endfunction
- In this case the debugger in single step mode will highlight line 1 after
- line 3 whereas this won't happen for:
-function test
- a=1
endfunction
+ The debugger in single step mode highlights line 1 after line 2 whereas
+ line 2 is the last line of the function source code
+ tree2code(macrc2tree(test)) has 4 lines (including 2 empty lines) while
+ fun2string(test) contains the correct output with only 2 lines

```

```
-watching some very large variable content looks like the variable is empty
This is Tk bug 1449858.
@@ -532,7 +519,13 @@
```

Misc notes:

```
-- Could find a use in the Scipad code (all are Tcl/Tk 8.5 features):
+- Tcl/Tk 8.5 features that could find their use in the Scipad code:
  font actual $font $char (to know whether $char is included in $font)
  text count
  text search -all
+
+- Tcl/Tk 8.6 features that could find their use in the Scipad code:
+ file tempfile (TIP #210)
+ tk busy (TIP #321)
+ tk fontchooser (TIP #324)
+ -insertunfocussed for text widgets (TIP #197)
--- changelog.txt      Tue Jan 06 22:15:00 2009
+++ changelog.txt      Sun Jan 11 11:07:36 2009
@@ -1,3 +1,10 @@
+Francois VOGEL, 11/01/09
+ * Selection of the file encoding is now possible through a new option menu,
+   with an MUR list of encodings, and auto-detection of encoding for XML files
+   (this is a request from Yung-Jang Lee)
+   This is the implementation corresponding to SEP#12_V2.1
+ * version --> 7.14
+
+Francois VOGEL, 06/01/09
+ * Fixed Scipad part of bug 3407 (Scilab 5 returns physical line numbers in
+   lasterror while Scilab 4 and Scilab-gtk return logical line numbers)
@@ -100,7 +107,7 @@
+ * Fixed bug 3416 (black Scipad icon on Windows 2000) again since the
+   candidate fix in 6.156 didn't work
+ * version --> 6.157 - This version comes with Scilab 5.0 final release,
-   Scilab 5.0.1 and Scilab 5.0.2
+   Scilab 5.0.1, Scilab 5.0.2 and Scilab 5.0.3

Francois VOGEL, 05/09/08
+ * Fixed bug 3416 (black Scipad icon on Windows 2000)
--- tcl/db_bptsgui.tcl Tue Jan 06 00:45:32 2009
+++ tcl/db_bptsgui.tcl Sat Jan 10 11:58:26 2009
@@ -77,6 +77,13 @@
#       solution here. For the time being, this dialog is not even
#       translated and the debugger is broken (bug 2789), so let's wait.
#       Once this is done, proc mcmaxra should be erased since no longer used
+ #       A possible solution would be to save the currently selected item in
+ #       each spinbox, search for the longest item of the spinbox, select it,
+ #       grid the spinbox, select back the item initially selected. Searching
+ #       for the longest item is more precisely gridding the item that has
+ #       max{ [font measure $item $textFont] }
+ #       See also the end of the discussion here:
+ #       http://groups.google.fr/group/comp.lang.tcl/browse\_thread/thread/dfb592a3c9335b38
  set funcolwidth [mcmaxra "Function"]
  if {$funcolwidth < $maxcharinascilabname} {set funcolwidth $maxcharinascilabname}
  set lincolwidth [mcmaxra "Line"]
--- tcl/db_debugsession.tcl Tue Jan 06 00:45:32 2009
+++ tcl/db_debugsession.tcl Sat Jan 10 11:58:31 2009
@@ -48,6 +48,7 @@
global tmpdir
global previousstopfun
global bptsprops
+ global defaultencoding

  if {[isscilabbusy 5]} {return}
  showinfo $waitmessage
@@ -114,6 +115,7 @@
  if {[catch {
    set fname [file join $tmpdir "Scipad_execfile_bp_tempfile.sci"]
    set fid [open $fname w]
+   fconfigure $fid -encoding $defaultencoding
    puts $fid $allfuntxts
    close $fid
    ScilabEval_lt "exec(\"$fname\");" "sync" "seq"
--- tcl/defaults.tcl Tue Jan 06 21:24:58 2009
+++ tcl/defaults.tcl Sat Jan 10 11:59:02 2009
@@ -178,13 +178,13 @@
# those are the preferences which are going to be saved
set listofpref "$colorpref wordWrap \
  WMGEOMETRY WMSTATE printCommand indentspaces tabsizemargins \
- usekeywordindent \
```

```

-     filenamesdisplaytype maxrecentfiles scilabSingleQuotedStrings \
-     tabinserts lang completionbinding showContinuedLines \
-     filebackupdepth bindstyle doubleclickscheme colorizeenable \
-     windowmenusorting linenumbersmargins ScilabErrorMessageBox \
-     colorizeuserfuns showclosureXcross exitwhenlastclosed"
+set listofpref_list { listofrecent textFont menuFont }
+ usekeywordindent filenamesdisplaytype maxrecentfiles maxrecentencodings \
+ scilabSingleQuotedStrings tabinserts lang completionbinding \
+ showContinuedLines filebackupdepth bindstyle doubleclickscheme \
+ colorizeenable windowmenusorting linenumbersmargins \
+ ScilabErrorMessageBox colorizeuserfuns showclosureXcross \
+ exitwhenlastclosed autodetectencodinginxmlfiles"
+set listofpref_list { listofrecentfiles listofrecentencodings textFont menuFont }

# default options which can be overridden
set wordWrap "none"
@@ -220,7 +220,9 @@
set usekeywordindent 1 ; # use smart keyword indentation: 0 (no) or 1 (yes)
set filenamesdisplaytype "pruned" ;# "pruned" or "full" or "fullifambig"
set maxrecentfiles 15
-set listofrecent [list] ;# always full filenames here
+set maxrecentencodings 5
+set listofrecentfiles [list] ;# always full filenames here
+set listofrecentencodings [list]
set scilabSingleQuotedStrings "yes"
set tabinserts "spaces" ;# "spaces" or "tabs"
set completionbinding "Control-Tab"
@@ -235,6 +237,7 @@
set colorizeuserfuns "yes"
set showclosureXcross true
set exitwhenlastclosed false
+set autodetectencodinginxmlfiles true

# End of saved preferences
#####
@@ -296,6 +299,10 @@
# frame pathname in which $textarea is packed, or "none" if it is not packed
array unset pwframe

+# default encoding is the system native encoding
+set defaultencoding [encoding system]
+set currentencoding $defaultencoding
+
#####
# source the user preferences file if any
# this must happen after the locale selection from Scilab's getlanguage() above
@@ -604,6 +611,14 @@
# Scilab matrix of strings or string (all with no continuation dots nor comments)
set ssmsRE {}
append ssmsRE {(?:) $smstRE_rep {}|(?:) $sstrRE {}}
+
+###
+
+# regular expression matching the start of an XML prolog and reporting
+# an encoding name in that prolog
+# see the XML specification http://www.w3.org/TR/xml
+# see also use of this RE in proc detectencoding
+set xml_prologstart_RE_rep {<?xml[[:blank:]]+version[[:blank:]]*=[[:blank:]]*["']1.[[:digit:]]+
["'][:blank:]]+encoding[[:blank:]]*=[[:blank:]]*["']([[:alpha:]][\w.-]+)["']}

###

--- tcl/filecommands.tcl      Tue Jan 06 00:45:32 2009
+++ tcl/filecommands.tcl      Sun Jan 11 11:34:21 2009
@@ -111,6 +111,11 @@
#     true: file gets colored
#     false: no colorization for this file
#
+#     listoffile("$ta",encoding)
+#     name of the encoding in which the file is stored,
+#     e.g. utf-8 or euc-jp or cpl252 or a lot of other possibilities
+#     See SEP#12 for a complete description of the encodings support in Scipad
+#
#     The windows menu entries are radionbuttons, with the following
#     properties:
#     -value is $winopened
@@ -135,6 +140,7 @@
proc filesetasnew {} {
    global winopened listoffile
    global listoftextarea pad

```

```

+   global defaultencoding

       # ensure that the cursor is changed to the default cursor
       event generate [gettextarea cur] <Leave>
@@ -151,6 +157,7 @@
       set listoffile("$pad.new$winopened",undostackdepth) 0
       set listoffile("$pad.new$winopened",redostackdepth) 0
       set listoffile("$pad.new$winopened",progressbar id) ""
+   set listoffile("$pad.new$winopened",encoding) $defaultencoding
       lappend listoftextarea $pad.new$winopened

       addwindowsmenuentry $winopened $listoffile("$pad.new$winopened",displayedname)
@@ -341,6 +348,7 @@
       unset listoffile("$textarea",undostackdepth)
       unset listoffile("$textarea",redostackdepth)
       unset listoffile("$textarea",progressbar_id)
+   unset listoffile("$textarea",encoding)

       # the rest of this proc is similar to proc hidetext,
       # but not identical
@@ -620,7 +628,7 @@
       # but unfortunately this does not work when used from the debugger
       # call stack area (the shell goes deeper one level and execution is
       # delayed), therefore a more complicated solution here

-
+   set fullcomm
"TCL_EvalStr(\openfile \\\\"+strsubst(get_function_path(\$nametooopen\\),\\"\\\",\\"/\")
+\\\"\\\"\\\",\\scipad\");"
+   set fullcomm
"TCL_EvalStr(\openfile \\\\"+strsubst(get_function_path(\$nametooopen\\),\\"\\\",\\"/\")+\\\"\\\"
currenttile iso8859-1\\\",\\scipad\");"
       ScilabEval_lt $fullcomm "seq"
       }
       "scicos" {
@@ -633,7 +641,7 @@
       # <TODO>: the line below does not need the ScilabEval(TCL_EvalStr...) construct
       # because there is no Scilab instruction inside
       # openfile $filetoopen should be just enough - to be checked
-   ScilabEval_lt "TCL_EvalStr(\openfile \\\\"$filetoopen\\\"\\\",\\scipad\");" "seq"
+   ScilabEval_lt "TCL_EvalStr(\openfile \\\\"$filetoopen\\\"\\\" currenttile
iso8859-1\\\",\\scipad\");" "seq"
       }
       "userfun" {
       set nameinitial [string range $nametooopen 0 0]
@@ -760,19 +768,20 @@
       }
       }

-proc openfile {file {tiledisplay "currenttile"}} {
+proc openfile {file {tiledisplay "currenttile"} {encodingtouse ""}} {
       # try to open a file with filename $file (no file selection through a dialog)
-# if file is not already open, open it
+# if file is not already open, open it using encoding $encodingtouse
       # otherwise just switch buffers to show it
       # return value:
       # 0 if file could not be open
       # 1 if file could be open or displayed (switched buffers)
       global pad winopened listoftextarea listoffile
       global closeinitialbufferallowed startdir
+   global currentencoding

-#hack for bringing up the chooser, if $file is a directory
-# on windows this has to precede the check for readable,
-# because a directory is "unreadable"
+   # hack for bringing up the chooser, if $file is a directory
+   # on windows this has to precede the check for readable,
+   # because a directory is "unreadable"
       if {[file isdirectory $file]} {
           set startdir $file
           showopenwin currenttile;
@@ -793,7 +802,7 @@
       if {[file exists $file]} {
           set listoffile("$pad.new$winopened",thetime) [file mtime $file]
           set listoffile("$pad.new$winopened",new) 0
           shownewbuffer $file $tiledisplay
+   # unless otherwise specified, use the encoding selected in the options/encoding
menu
+   if {$encodingtouse eq ""} {
+       set encodingtouse $currentencoding
+   }
           shownewbuffer $file $tiledisplay $encodingtouse

```



```

+         # update the options menu and encoding property in listoffile
+         set currentencoding $encodingtouse
+         setencoding
+     } else {
+         set listoffile("$pad.new$winopened",thetime) 0
+         set listoffile("$pad.new$winopened",new) 1
@@ -839,6 +855,8 @@
# $file is not opened - this sets the $listoffile area values for that file
# and adds an entry in the windows menu
global winopened pad listoffile
+ global currentencoding autodetectencodinginxmlfiles
+
+ incr winopened
dupWidgetOption [gettextareacur] $pad.new$winopened
set listoffile("$pad.new$winopened",fullname) [file normalize $file]
@@ -856,13 +874,24 @@
set listoffile("$pad.new$winopened",redostackdepth) 0
set listoffile("$pad.new$winopened",progressbar_id) ""

+ if {$autodetectencodinginxmlfiles} {
+     # automatic detection of encoding (for xml files)
+     set detenc [detectencoding $file]
+     set listoffile("$pad.new$winopened",encoding) $detenc
+     set currentencoding $detenc
+     setencoding
+ } else {
+     # no auto-detection of encoding, just use the current encoding
+     # selected in the options/encoding menu
+     set listoffile("$pad.new$winopened",encoding) $currentencoding
+ }
+ addwindowsmenuentry $winopened $listoffile("$pad.new$winopened",displayedname)
+ }

-proc shownewbuffer {file tiledisplay} {
+proc shownewbuffer {file tiledisplay encodingtouse} {
global pad winopened closeinitialbufferallowed
if {[fileunreadable $file]} {return}
- openfileondisk $pad.new$winopened $file $tiledisplay
+ openfileondisk $pad.new$winopened $file $tiledisplay $encodingtouse
resetmodified $pad.new$winopened "clearundoredo stacks"
# colorization must be launched before showing the textarea
# so that foreground colorization while stepping into
@@ -896,7 +925,7 @@
file to another name and reopen it from disk!" ] -parent $pad
}

-proc openfileondisk {textarea thefile tiledisplay} {
+proc openfileondisk {textarea thefile tiledisplay encodingtouse} {
# really open/read a file from disk
# all readability tests have normally been done before
global listoftextarea pad closeinitialbufferallowed
@@ -908,9 +937,8 @@
closefile $pad.new1
}
set newnamefile [open $thefile r]
- while {[eof $newnamefile]} {
-     $textarea insert end [read -nonewline $newnamefile ]
- }
+ fconfigure $newnamefile -encoding $encodingtouse
+ $textarea insert end [read -nonewline $newnamefile]
close $newnamefile
}

@@ -1138,7 +1166,7 @@
# really write the file onto the disk
# all writability tests have normally been done before
global filebackupdepth tcl_platform
- global pad
+ global pad listoffile

if {$snobackupskip} {
backupfile $nametosave $filebackupdepth
@@ -1187,6 +1215,7 @@
}

set FileNameToSave [open $nametosave w]
+ fconfigure $FileNameToSave -encoding $listoffile("$textarea",encoding)
puts -nonewline $FileNameToSave [$pad.temptextwidget get 1.0 end]
close $FileNameToSave

```

```

@@ -1199,9 +1228,14 @@

proc savepreferences {} {
    global env listofpref listofpref_list
+   global defaultencoding
    set preffilename [file join $env(SCIHOME) .SciPadPreferences.tcl]
    catch {
        set preffile [open $preffilename w]
+       # the preferences file is always saved with the default platform system encoding
+       # this avoids to source it later in an encoding different than the platform
+       # native encoding, which is a feature available only from Tcl 8.5 onwards
+       fconfigure $preffile -encoding $defaultencoding
        foreach opt $listofpref {
            global $opt
            puts $preffile [concat "set $opt" \{[set $opt]\}]
@@ -1271,6 +1305,7 @@
        }
        if {$answer == yes} {
            set oldfile [open $thefile r]
+           fconfigure $oldfile -encoding $listoffile("$textarea",encoding)
            $textarea delete 1.0 end
            while {[eof $oldfile]} {
                $textarea insert end [read -nonewline $oldfile ]
@@ -1333,6 +1368,107 @@
    }

#####
+# file encoding procs
+#####
+proc setencoding {} {
+# set the encoding property of the current buffer to be the encoding
+# currently selected in the encoding options menu
+# this proc is called when selecting any option of this menu
+   global currentencoding listoffile
+
+   set textarea [gettextareacur]
+   foreach ta [getfullpeerset $textarea] {
+       set listoffile("$ta",encoding) $currentencoding
+       modifiedtitle $ta
+   }
+
+   # encoding system MUST NOT be modified because the system encoding
+   # for the platform on which Scipad is running does not change
+   # Saying encoding system $currentencoding here would result in a
+   # number of issues such as encoding eating $env:
+   # can't read "env(SCIHOME)": no such variable
+   # This would happen in a number of situations and can be trimmed down to:
+   #   % encoding system
+   #   cp1252
+   #   % puts $env(TEMP)
+   #   C:\DOCUME~1\FRANOI~1\LOCALS~1\Temp
+   #   % encoding system gb2312-raw
+   #   % puts $env(TEMP)
+   #   can't read "env(TEMP)": no such variable
+   #   % parray env
+   #   missing close-brace
+   # see also: http://groups.google.fr/group/comp.lang.tcl/browse\_thread/thread/ef4676cee655d97
+
+   # add in the MRU list of encodings
+   AddRecentEncoding $currentencoding
+}
+
+proc detectencoding {filename} {
+# detect encoding of $filename if it is an xml file
+# this is done by looking for the encoding name in the prolog
+# of the file
+# if detection fails for any reason, the current encoding name is returned
+# if detection succeeds, then the returned string it is an encoding name
+# among the list of known encodings
+
+   global xml_prologstart_RE_rep currentencoding
+
+   # check that the filename extension is xml
+   if {[extenstolang $filename] ne "xml"} {
+       return $currentencoding
+   }
+
+   # extension is xml, read the file using the current encoding
+   # if anything fails then return $currentencoding
+   if {[catch { \
            set fileid [open $filename r]; \

```

```

+         fconfigure $fileid -encoding $currentencoding; \
+         set allthetext [read -nonewline $fileid]; \
+         close $fileid \
+     } \
+ }} {
+     return $currentencoding
+ }
+
+ # regexp match the encoding name in the prolog
+ set encodingwasfound [regexp -nocase $xml_prologstart_RE_rep $allthetext -> detectedencoding]
+ if {$encodingwasfound} {
+     # encoding name found, now contained in $detectedencoding
+     # the list of encoding is apparently returned in lower case by [encoding names]
+     # therefore the detected encoding name really is the lowercase conversion
+     # of what was found in the xml prolog
+     set detectedencoding [string tolower $detectedencoding]
+     if {[lsearch -exact [encoding names] $detectedencoding] != -1} {
+         # detected encoding name is known by Scipad
+         return $detectedencoding
+     } else {
+         # Scipad (in fact Tcl) is not aware of the detected encoding
+         # try to remove the dash after iso and check again since
+         # iso8859-1 is part of [encoding system] while iso-8859-1 is mentioned
+         # in Scilab xml help files
+         set detectedencoding2 [string map {iso- iso} $detectedencoding]
+         if {[lsearch -exact [encoding names] $detectedencoding2] != -1} {
+             # detected encoding name is now known by Scipad
+             return $detectedencoding2
+         } else {
+             # try to add a dash after utf and check again since
+             # utf-8 is part of [encoding system] while utf8 might be used
+             # in some xml help files although this is incorrect
+             set detectedencoding2 [string map {utf utf-} $detectedencoding]
+             if {[lsearch -exact [encoding names] $detectedencoding2] != -1} {
+                 # detected encoding name is now known by Scipad
+                 return $detectedencoding2
+             } else {
+                 return $currentencoding
+             }
+         }
+     }
+ } else {
+     # no match in the xml prolog
+     return $currentencoding
+ }
+}
+
+#####
+ # ancillaries for file commands
+#####
+proc getpathandext {ta} {
@@ -1881,134 +2017,5 @@
+     set newname [file join $newname $tojoin]
+ }
+     set listoffile("$ta",displayedname) $newname
+ }
+}
+
+#####
+ -# procedures dealing with the recent files list
+ -# displayed in the file menu
+#####
+ -proc AddRecentFile {filename} {
+ -# add a new recent file item to the file menu
+ -# if there is already the max number of entries, then shift them
+ -# one line down and insert $filename at the top
+ - global pad listofrecent maxrecentfiles nbrecentfiles
+ - if {$maxrecentfiles == 0} {return}
+ - # first check if the new entry is already present
+ - set present "false"
+ - for {set i 0} {$i<$nbrecentfiles} {incr i} {
+ -     if {[lindex $listofrecent $i] == $filename} {
+ -         set present "true"
+ -         set pospresent $i
+ -     }
+ - }
+ - set reclind [GetFirstRecentInd]
+ - if {$present == "false"} {
+ -     # add the new entry
+ -     if {$nbrecentfiles == 0} {

```



```

global pad winTitle ScipadVersion listoffile
global MenuEntryId
- set fname $listoffile("$textarea",displayedname)
- set ind [extractindexfromlabel $pad.filemenu.wind $fname]
- set mod1 ""; set mod2 ""
+ global defaultencoding
+
+ if {$listoffile("$textarea",readonly) == 1} {
+     set mod1 [mc " \[ReadOnly\]"]
+ } else {
+     set mod1 ""
+ }
+
+ if {[isanymodified]} {
+     $pad.statusind configure -background PeachPuff
+ } else {
+     $pad.statusind configure -background [$pad.filemenu cget -background]
+ }
+
+ set fname $listoffile("$textarea",displayedname)
+ set ind [extractindexfromlabel $pad.filemenu.wind $fname]
+ if {[ismodified $textarea]} {
+     set mod2 [mc " (modified)"]
+     if {$ind !=-1} {
@@ -199,24 +205,34 @@
+     }
+     $pad.statusind configure -background Salmon
+ } else {
+     set mod2 ""
+     if {$ind !=-1} {
+         $pad.filemenu.wind entryconfigure $ind -background "" \
+             -activebackground ""
+     }
+ }
+
+ if {$listoffile("$textarea",encoding) != $defaultencoding} {
+     set mod3 " ($listoffile("$textarea",encoding))"
+ } else {
+     set mod3 ""
+ }
+
+ if {$panesonly == "false"} {
+     # caught because scan will fail when launched from wish
+     if {[catch {
+         scan $ScipadVersion "%s - %s" ScipadVersionNumber ScipadVersionString
- wm title $pad "$winTitle $ScipadVersionNumber - $fname$mod1$mod2"
+ wm title $pad "$winTitle $ScipadVersionNumber - $fname$mod1$mod2$mod3"
+     }]} {
- wm title $pad "$winTitle - $fname$mod1$mod2"
+ wm title $pad "$winTitle - $fname$mod1$mod2$mod3"
+     }
+ }
+
+ if {[isdisplayed $textarea]} {
+     [getpaneframename $textarea].panetitle configure \
-     -text "$fname$mod1$mod2"
+     -text "$fname$mod1$mod2$mod3"
+ }
+
+ if {[ismodified $textarea] && \
+     $listoffile("$textarea",thetime) !=0} {
+     $pad.filemenu.files entryconfigure \
--- tcl/mainwindow.tclTue Jan 06 00:45:32 2009
+++ tcl/mainwindow.tclSat Jan 10 12:01:08 2009
@@ -23,6 +23,7 @@
+ #
+ # See the file scipad/license.txt
+ #
+
+ toplevel $pad
+ setscipadicon $pad
+
@@ -42,6 +43,7 @@
+ set listoffile("$pad.new$winopened",undostackdepth) 0; # used to enable/disable the undo menu entry
+ set listoffile("$pad.new$winopened",redostackdepth) 0; # used to enable/disable the redo menu entry
+ set listoffile("$pad.new$winopened",progressbar_id) ""; # colorization progressbar identifier
+set listoffile("$pad.new$winopened",encoding) $defaultencoding
+
+ set chset() {}
+ set words() {}

```

```

--- Makefile.am      Mon Jan 05 21:33:07 2009
+++ Makefile.am      Sat Jan 10 12:01:15 2009
@@ -48,6 +48,7 @@
 tcl/mainwindow.tcl \
 tcl/menues.tcl \
 tcl/modselection.tcl \
+tcl/MRUlist.tcl \
 tcl/platformbind.tcl \
 tcl/popupmenus.tcl \
 tcl/print.tcl \
--- tcl/menues.tcl   Tue Jan 06 00:45:32 2009
+++ tcl/menues.tcl   Sat Jan 10 12:01:32 2009
@@ -30,9 +30,12 @@
 global listoffile listoftextarea bindset
 global FirstBufferNameInWindowsMenu
 global FirstMRUFileNameInFileMenu
+ global FirstMRUEncodingInOptEncMenu
 global Shift_Tab
 foreach c1 "$bgcolors $fgcolors" {global $c1}
 global Tk85
+ global currentencoding defaultencoding nbrecentencodings
+ global autodetectencodinginxmlfiles

#destroy old menus (used when changing language)
foreach w [winfo children $pad.filemenu] {
@@ -358,12 +361,12 @@
     eval "$pad.filemenu.options.windmenusort add radiobutton \
         [me "&Most recently used"] -command {sortwindowsmenuentries} \
         -value MRUorder -variable windowsmenusorting"
- set recentnumbmenu [tk_optionMenu $pad.filemenu.options.recent \
+ set recentfilesnumbmenu [tk_optionMenu $pad.filemenu.options.recentf \
         maxrecentfiles 0 1 2 3 4 5 6 7 8 9 10 15 20 50]
     eval "$pad.filemenu.options add cascade [me "&Recent files"] \
         -menu $recentnumbmenu"
- for {set i 0} {$i<=[recentnumbmenu index last]} {incr i} {
-     $recentnumbmenu entryconfigure $i -command {UpdateRecentFilesList}
+ for {set i 0} {$i<=[recentfilesnumbmenu index last]} {incr i} {
+     $recentfilesnumbmenu entryconfigure $i -command {UpdateRecentFilesList}
 }
     eval "$pad.filemenu.options add cascade [me "&Backup files depth"] \
         -menu [tk_optionMenu $pad.filemenu.options.backup \
@@ -380,7 +383,7 @@
         -variable lang -value $l -command relocate"
 }
-# feature temporary disabled as not yet 100% ok (see bindings/issues.txt)
+# feature enabled yet not 100% ok (see bindings/issues.txt) - teasing!
 menu $pad.filemenu.options.bindings -tearoff 0
 eval "$pad.filemenu.options add cascade [me "&Bindings style"] \
     -menu $pad.filemenu.options.bindings "
@@ -405,6 +408,34 @@
     eval "$pad.filemenu.options.messageboxes add radiobutton \
         [me "Copied in a &message box"] \
         -value true -variable ScilabErrorMessageBox"
+ eval "$pad.filemenu.options add cascade [me "Enc&oding"] \
+     -menu $pad.filemenu.options.encodings"
+ menu $pad.filemenu.options.encodings -tearoff 1
+ set FirstMRUEncodingInOptEncMenu [expr {[ $pad.filemenu.options.encodings index last ] + 1}]
+ BuildInitialRecentEncodingsList
+ # if the MRU list of encodings is empty, then add platform system encoding, iso8859-1 and
utf-8
+ # these encodings are supposed to exist (distributed with Tcl)
+ if {$nbrecentencodings == 0} {
+     AddRecentEncoding "utf-8"
+     AddRecentEncoding "iso8859-1"
+     AddRecentEncoding $defaultencoding
+ }
+ eval "$pad.filemenu.options.encodings add cascade [me "More enc&odings"] \
+     -menu $pad.filemenu.options.encodings.more"
+ menu $pad.filemenu.options.encodings.more -tearoff 1
+ foreach en [lsort -dictionary [encoding names]] {
+     $pad.filemenu.options.encodings.more add radiobutton -label $en \
+         -command {setencoding} -value $en -variable currentencoding
+ }
+ set recentencodingsnumbmenu [tk_optionMenu $pad.filemenu.options.encodings.recente \
+     maxrecentencodings 0 1 2 3 4 5 6 7 8 9 10]
+ eval "$pad.filemenu.options.encodings add cascade [me "&Recent encodings"] \
+     -menu $recentencodingsnumbmenu"
+ for {set i 0} {$i<=[recentencodingsnumbmenu index last]} {incr i} {

```

```

+         $recentencodingsnumbmenu entryconfigure $i -command {UpdateRecentEncodingsList}
+     }
+     eval "$pad.filemenu.options.encodings add check [me "&Auto-detect encoding when loading XML
files"] \
+         -offvalue false -onvalue true -variable autodetectencodinginxmlfiles"
+     menu $pad.filemenu.options.exitopts -tearoff 0
+     eval "$pad.filemenu.options add cascade [me "E&xit options"] \
+         -menu $pad.filemenu.options.exitopts "
@@ -497,7 +528,7 @@
+     global Tk85
+     set tileprocalreadyrunning true
+     # File/Close
+     set iClose [expr {[GetFirstRecentInd] + $nbrecentfiles + 1}]
+     set iClose [expr {[GetFirstRecentFileInd] + $nbrecentfiles + 1}]
+     $pad.filemenu.files entryconfigure $iClose -state disabled
+     binddisable $pad {closecurfile yesnocancel}
+     # Windows menu entries
@@ -527,7 +558,7 @@
+     global tileprocalreadyrunning
+     global Tk85
+     # File/Close
+     set iClose [expr {[GetFirstRecentInd] + $nbrecentfiles + 1}]
+     set iClose [expr {[GetFirstRecentFileInd] + $nbrecentfiles + 1}]
+     $pad.filemenu.files entryconfigure $iClose -state normal
+     bindenable $pad {closecurfile yesnocancel}
+     # Windows menu entries
@@ -772,13 +803,13 @@
+     proc showinfo_menu_file {w} {
+     # display full pathname of a recent file entry of the file menu
+     # as a showinfo
+     -     global pad nbrecentfiles listofrecent
+     +     global pad nbrecentfiles listofrecentfiles
+     +     if {$nbrecentfiles > 0} {
+     -         set reclind [GetFirstRecentInd]
+     +         set reclind [GetFirstRecentFileInd]
+         set recnind [expr {$reclind + $nbrecentfiles - 1}]
+         set mouseentry [$w index active]
+         if {$reclind<=$mouseentry && $mouseentry<=$recnind} {
+     -         showinfo [lindex $listofrecent [expr {$mouseentry - $reclind}]]
+     +         showinfo [lindex $listofrecentfiles [expr {$mouseentry - $reclind}]]
+     }
+     }
+ }
+ }
--- tcl/print.tcl      Tue Jan 06 00:45:33 2009
+++ tcl/print.tcl      Sat Jan 10 12:01:40 2009
@@ -94,6 +94,7 @@
+     if {[ismodified $textarea] ||
+         ![file exists $listoffile("$textarea",fullname)]} {
+     +     set TempPrintFile [open /tmp/SciPadtmpfile w]
+     +     fconfigure $TempPrintFile -encoding $listoffile("$textarea",encoding)
+     +     puts -nonewline $TempPrintFile [$textarea get 1.0 end]
+     +     close $TempPrintFile
+     +     catch {eval exec "$sprintCommand /tmp/SciPadtmpfile"} result
@@ -120,6 +121,7 @@
+     ![file exists $listoffile("$textarea",fullname)]} {
+     +     set fname [file join $tmpdir SciPadtmpfile]
+     +     set TempPrintFile [open $fname w]
+     +     fconfigure $TempPrintFile -encoding $listoffile("$textarea",encoding)
+     +     puts -nonewline $TempPrintFile [$textarea get 1.0 end]
+     +     close $TempPrintFile
+     +     ScilabEval_lt "toprint(\"$fname\")" "sync"
--- tcl/scilabexec.tcl Tue Jan 06 22:16:09 2009
+++ tcl/scilabexec.tcl Sat Jan 10 12:01:51 2009
@@ -366,6 +366,7 @@
+     global tmpdir
+     global pad
+     +     global defaultencoding
+
+     +     set bsiz_1 4095 ;# Must be consistent with bsiz defined in stack.h
+     +     set lsiz_1 65535 ;# Must be consistent with lsiz defined in stack.h
@@ -384,6 +385,7 @@
+     +     set splitsize 4000 ;# arbitrary but works up to approx. 4095
+     +     set nbparts [expr {[string length $comm] / $splitsize + 1}]
+     +     set fid [open $fname w]
+     +     fconfigure $fid -encoding $defaultencoding
+     +     set startpos 0
+     +     for {set i 1} {$i < $nbparts} {incr i} {
+     +         set stoppos [expr {$i * $splitsize - 1}]
--- tcl/scipad.tcl      Mon Jan 05 21:33:07 2009

```



```

+++ tcl/scipad.tcl      Sat Jan 10 12:01:54 2009
@@ -90,6 +90,7 @@
    source [file join $sourcedir tooltips.tcl]
    source [file join $sourcedir progressbar.tcl]
    source [file join $sourcedir scrollableframe.tcl]
+   source [file join $sourcedir MRUlist.tcl]

    # now all the pure main level code
    source [file join $sourcedir defaults.tcl]
--- tcl/scipaddebug1.tcl      Mon Jan 05 21:33:07 2009
+++ tcl/scipaddebug1.tcl      Sat Jan 10 12:02:01 2009
@@ -296,6 +296,7 @@
    # and this one is never erased by Scipad (a new filename is used for each
    # bgererror)
    global Scipaddebuglogfileid Scipaddebuglogfileid env debuglog loginafilename
+   global defaultencoding
    if {(!$debuglog)} {return}
    timestamp sec mil
    puts "[clock format $sec -format "%d/%m/%y%T"].[format %03d $mil]|$value"
@@ -308,6 +309,7 @@
    } else {
        set Scipaddebuglogfileid [open $Scipaddebuglogfileid a]
    }
+   fconfigure $Scipaddebuglogfileid -encoding $defaultencoding
    puts $Scipaddebuglogfileid "[clock format $sec -format "%d/%m/%y%T"].[format %03d
$mil]|$value"
    close $Scipaddebuglogfileid
}
}
--- tcl/scipaddebug2.tcl      Mon Jan 05 21:33:07 2009
+++ tcl/scipaddebug2.tcl      Sat Jan 10 12:02:05 2009
@@ -45,7 +45,7 @@
    mc amp mcra mcmax mcmaxra bl \
    keyposn dokeyposn ismodified whichfun modifiedtitle \
    showinfo delinfo showinfo_menu_file showinfo_menu_wind \
-   GetFirstRecentInd extractindexfromlabel \
+   GetFirstRecentFileInd GetFirstRecentEncodingInd extractindexfromlabel \
    checkifanythingchangedondisk checkiffilechangedondisk \
    TextStyles managescroll \
    schememenus setdbmenuentriesstates bp getdbstate pbind \
--- version.xml      Tue Jan 06 21:34:07 2009
+++ version.xml      Sun Jan 11 11:35:12 2009
@@ -15,5 +15,5 @@
-->
<!-- ===== -->

-<VERSION major="7" minor="13" maintenance="0" revision="0" string="SciPad unstable-svn" />
+<VERSION major="7" minor="14" maintenance="0" revision="0" string="SciPad unstable-svn" />
</MODULE_VERSION>

```

File MRUlist.tcl to be placed in SCI/modules/scipad/tcl:

```

# Scipad - programmer's editor and debugger for Scilab
#
# Copyright (C) 2002 - INRIA, Matthieu Philippe
# Copyright (C) 2003-2006 - Weizmann Institute of Science, Enrico Segre
# Copyright (C) 2004-2008 - Francois Vogel
#
# Localization files ( in tcl/msg_files/) are copyright of the
# individual authors, listed in the header of each file
#
# This program is free software; you can redistribute it and/or modify
# it under the terms of the GNU General Public License as published by
# the Free Software Foundation; either version 2 of the License, or
# (at your option) any later version.
#
# This program is distributed in the hope that it will be useful,
# but WITHOUT ANY WARRANTY; without even the implied warranty of
# MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
# GNU General Public License for more details.
#
# You should have received a copy of the GNU General Public License
# along with this program; if not, write to the Free Software
# Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
#
# See the file scipad/license.txt
#

```

```

#####
# generic procedures dealing with a MRU (most recently used) list
#####

# <TODO> would be nice to factorize MRU lists procedures below!

#####
# procedures dealing with the recent files list
# displayed in the file menu
#####
proc AddRecentFile {filename} {
# add a new recent file item to the file menu
# if there is already the max number of entries, then shift them
# one line down and insert $filename at the top
    global pad listofrecentfiles maxrecentfiles nbrecentfiles
    if {$maxrecentfiles == 0} {return}
    # first check if the new entry is already present
    set present "false"
    for {set i 0} {$i<$nbrecentfiles} {incr i} {
        if {[lindex $listofrecentfiles $i] == $filename} {
            set present "true"
            set pospresent $i
        }
    }
    set reclind [GetFirstRecentFileInd]
    if {$present == "false"} {
        # add the new entry
        if {$nbrecentfiles == 0} {
            incr reclind
            $pad.filemenu.files insert $reclind separator
        }
        # update the file menu
        if {$nbrecentfiles < $maxrecentfiles} {
            incr nbrecentfiles
            # insert new entry
            set listofrecentfiles [linsert $listofrecentfiles 0 $filename]
            # [list [lindex $listofrecentfiles $i]] automatically escapes special characters
            $pad.filemenu.files insert $reclind command \
                -label [file tail [lindex $listofrecentfiles 0] ] \
                -command "openfileifexists [list [lindex $listofrecentfiles 0]]"
            # update menu entries (required to update the numbers)
            UpdateRecentLabels $reclind
        } else {
            # forget last entry of the list and insert new entry
            set listofrecentfiles [lreplace $listofrecentfiles end end]
            set listofrecentfiles [linsert $listofrecentfiles 0 $filename]
            # update menu entries
            UpdateRecentLabels $reclind
        }
    } else {
        # move the existing entry to the top of the list
        set listofrecentfiles [lreplace $listofrecentfiles $pospresent $pospresent]
        set listofrecentfiles [linsert $listofrecentfiles 0 $filename]
        UpdateRecentLabels $reclind
    }
}

proc GetFirstRecentFileInd {} {
# get index of first recent file item in the file menu
    global FirstMRUFileNameInFileMenu nbrecentfiles
    if {$nbrecentfiles == 0} {
        return [expr {$FirstMRUFileNameInFileMenu - 1}]
    } else {
        return $FirstMRUFileNameInFileMenu
    }
}

proc UpdateRecentLabels {reclind} {
# update labels of recent files entries with file tail preceded by a number
    global pad listofrecentfiles nbrecentfiles
    for {set i 0} {$i<$nbrecentfiles} {incr i} {
        if {$i<9} {
            set lab [concat [expr {$i + 1}] [file tail [lindex $listofrecentfiles $i] ] ]
        } else {
            set lab [file tail [lindex $listofrecentfiles $i] ]
        }
        set ind [expr {$reclind + $i}]
        # [list [lindex $listofrecentfiles $i]] automatically escapes special characters
        $pad.filemenu.files entryconfigure $ind \

```

```

        -label $lab \
        -command "openfileifexists [list [lindex $listofrecentfiles $i]]"
    if {$i<9} {
        $pad.filemenu.files entryconfigure $ind -underline 0
    } else {
        $pad.filemenu.files entryconfigure $ind -underline -1
    }
}
}

proc BuildInitialRecentFilesList {} {
    global pad listofrecentfiles nbrecentfiles
    set nbrecentfiles [llength $listofrecentfiles]
    for {set i 0} {$i<$nbrecentfiles} {incr i} {
        if {$i<9} {
            set lab [concat [expr {$i + 1}] [file tail [lindex $listofrecentfiles $i] ] ]
        } else {
            set lab [file tail [lindex $listofrecentfiles $i] ]
        }
        # [list [lindex $listofrecentfiles $i]] automatically escapes special characters
        $pad.filemenu.files add command \
            -label $lab \
            -command "openfileifexists [list [lindex $listofrecentfiles $i]]"
        if {$i<9} {
            set ind [$pad.filemenu.files index end]
            $pad.filemenu.files entryconfigure $ind \
                -underline 0
        }
    }
    if {$nbrecentfiles > 0} {
        $pad.filemenu.files add separator
    }
}

proc UpdateRecentFilesList {} {
    # this proc gets called whenever the maximum number of recent entries
    # in the file menu is changed by the user
    global pad listofrecentfiles maxrecentfiles nbrecentfiles
    if {$maxrecentfiles >= [llength $listofrecentfiles]} {
        # nothing to do, maxrecentfiles was just increased
        # this is handled by AddRecentFile
        return
    } else {
        # maxrecentfiles was decreased
        # forget the entries in listofrecentfiles, and update the menu
        set reclind [GetFirstRecentFileInd]
        set firstind [expr {$reclind + $maxrecentfiles}]
        set lastind [expr {$reclind + [llength $listofrecentfiles] - 1}]
        $pad.filemenu.files delete $firstind $lastind
        set listofrecentfiles [lreplace $listofrecentfiles $maxrecentfiles end]
        incr nbrecentfiles [expr {- ($lastind - $firstind + 1)}]
        if {$maxrecentfiles == 0} {
            # remove the separator
            $pad.filemenu.files delete $firstind
        }
    }
}

#####
# procedures dealing with the recent encodings list
# displayed in the options/encoding menu
#####
proc AddRecentEncoding {encod} {
    # add a new recent encoding item to the options/encoding menu
    # if there is already the max number of entries, then shift them
    # one line down and insert $encod at the top
    global pad listofrecentencodings maxrecentencodings nbrecentencodings
    if {$maxrecentencodings == 0} {return}
    # do nothing if the encoding passed here does not exist
    # (should never happen, unless encoding files *.enc are missing
    # from the Tcl distribution)
    if {[lsearch -exact [encoding names] $encod] == -1} {
        return
    }
    # first check if the new entry is already present
    set present "false"
    for {set i 0} {$i<$nbrecentencodings} {incr i} {
        if {[lindex $listofrecentencodings $i] eq $encod} {
            set present "true"
            set pospresent $i
        }
    }
}

```

```

    }
}
set reclind [GetFirstRecentEncodingInd]
if {$present == "false"} {
    # add the new entry
    if {$nbrecentencodings == 0} {
        incr reclind
        $pad.filemenu.options.encodings insert $reclind separator
    }
    # update the file menu
    if {$nbrecentencodings < $maxrecentencodings} {
        incr nbrecentencodings
        # insert new entry
        set listofrecentencodings [linsert $listofrecentencodings 0 $encod]
        # [list [lindex $listofrecentencodings $i]] automatically escapes special characters
        $pad.filemenu.options.encodings insert $reclind radiobutton \
            -label [lindex $listofrecentencodings 0] \
            -command "setencoding" \
            -value [lindex $listofrecentencodings 0] -variable currentencoding
        # update menu entries (required to update the numbers)
        UpdateRecentEncodingsLabels $reclind
    } else {
        # forget last entry of the list and insert new entry
        set listofrecentencodings [lreplace $listofrecentencodings end end]
        set listofrecentencodings [linsert $listofrecentencodings 0 $encod]
        # update menu entries
        UpdateRecentEncodingsLabels $reclind
    }
} else {
    # move the existing entry to the top of the list
    set listofrecentencodings [lreplace $listofrecentencodings $pospresent $pospresent]
    set listofrecentencodings [linsert $listofrecentencodings 0 $encod]
    UpdateRecentEncodingsLabels $reclind
}
}

proc GetFirstRecentEncodingInd {} {
    # get index of first recent file item in the file menu
    global FirstMRUEncodingInOptEncMenu nbrecentencodings
    if {$nbrecentencodings == 0} {
        return [expr {$FirstMRUEncodingInOptEncMenu - 1}]
    } else {
        return $FirstMRUEncodingInOptEncMenu
    }
}

proc UpdateRecentEncodingsLabels {reclind} {
    # update labels of recent encodings entries preceded by a number
    global pad listofrecentencodings nbrecentencodings
    for {set i 0} {$i < $nbrecentencodings} {incr i} {
        if {$i < 9} {
            set lab [concat [expr {$i + 1}] [lindex $listofrecentencodings $i] ]
        } else {
            set lab [lindex $listofrecentencodings $i]
        }
        set ind [expr {$reclind + $i}]
        $pad.filemenu.options.encodings entryconfigure $ind \
            -label $lab \
            -command "setencoding" \
            -value [lindex $listofrecentencodings $i] -variable currentencoding
        if {$i < 9} {
            $pad.filemenu.options.encodings entryconfigure $ind -underline 0
        } else {
            $pad.filemenu.options.encodings entryconfigure $ind -underline -1
        }
    }
}

proc BuildInitialRecentEncodingsList {} {
    global pad listofrecentencodings nbrecentencodings
    set nbrecentencodings [llength $listofrecentencodings]
    for {set i 0} {$i < $nbrecentencodings} {incr i} {
        if {$i < 9} {
            set lab [concat [expr {$i + 1}] [lindex $listofrecentencodings $i] ]
        } else {
            set lab [lindex $listofrecentencodings $i]
        }
        $pad.filemenu.options.encodings add radiobutton \
            -label $lab \
            -command "setencoding" \

```

```

        -value [lindex $listofrecentencodings $i] -variable currentencoding
    if {$i<9} {
        set ind [$pad.filemenu.options.encodings index end]
        $pad.filemenu.options.encodings entryconfigure $ind \
            -underline 0
    }
}
if {$nbrecentencodings > 0} {
    $pad.filemenu.options.encodings add separator
}
}

proc UpdateRecentEncodingsList {} {
    # this proc gets called whenever the maximum number of recent entries
    # in the options/encodings menu is changed by the user
    global pad listofrecentencodings maxrecentencodings nbrecentencodings
    if {$maxrecentencodings >= [llength $listofrecentencodings]} {
        # nothing to do, maxrecentencodings was just increased
        # this is handled by AddRecentEncoding
        return
    } else {
        # maxrecentencodings was decreased
        # forget the entries in listofrecentencodings, and update the menu
        set reclind [GetFirstRecentEncodingInd]
        set firstind [expr {$reclind + $maxrecentencodings}]
        set lastind [expr {$reclind + [llength $listofrecentencodings] - 1}]
        $pad.filemenu.options.encodings delete $firstind $lastind
        set listofrecentencodings [lreplace $listofrecentencodings $maxrecentencodings end]
        incr nbrecentencodings [expr {- ($lastind - $firstind + 1)}]
        if {$maxrecentencodings == 0} {
            # remove the separator
            $pad.filemenu.options.encodings delete $firstind
        }
    }
}
}

```

Still <TODO>

- Check implications on bug 2520 (needs the debugger to work again first – bug 2789)
- Regenerate makefile.in since the new file MRUlist.tcl has been added in makefile.am

Changelog

- 1.0 – F. Vogel 15/12/08.
Initial version.
- 1.1 – F. Vogel 23/12/08.
Dealt with the case of Scipad internal files.
- 2.0 – F. Vogel 01/01/09.
Added an MRU list of encodings
Auto-detection of encoding in XML files.
- 2.1 – F. Vogel 11/01/09.
encoding system must not be changed by this SEP

Copyright

This document has been placed under the NOL.