EMACS Extensions

That you too might find useful

2012 Igor Lapitski, Hyde Park

Image from Wikipedia

> *"Emacs is not an editor. Emacs is a way of thinking about the world, and as such is a way of thinking about editors. When you ask what Emacs does, you are asking a question with no answer, because Emacs doesn't do, it is done to. Emacs just is ... I hope this makes things clearer." ~*
>
> *Scott Dorsey*

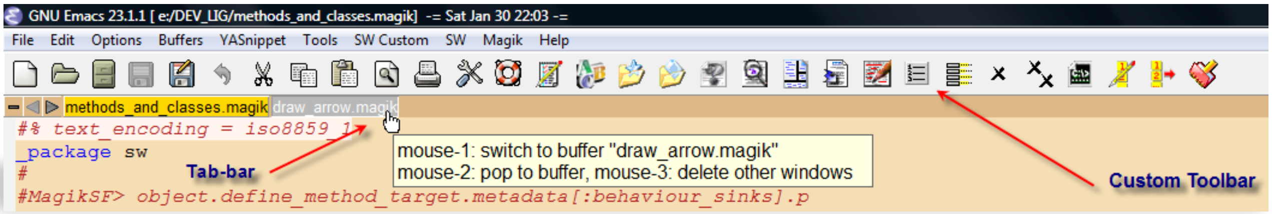
Why is it called – Emacs?
The name 'Emacs' was originally chosen as an abbreviation of **E**ditor **MAC**ro**S** and was created by Free Software Foundation.

<u>This document includes highlights of some new and existing features and fine points in this customised version of Emacs.</u>

This customised version based on the Smallworld additions to Emacs issued with GE Smallworld 4.0.

All comments and suggestions please forward to <igor@hydepark-consulting.com> . You can download this Emacs from www.hydepark-consulting.com/Emacs .

- Custom Toolbar and Tab-bar (use <Ctrl-Tab> to navigate between tabs).

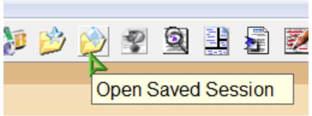


GNU Emacs 23.1.1 [e:/DEV_LIG/methods_and_classes.magik] -= Sat Jan 30 22:03 -=

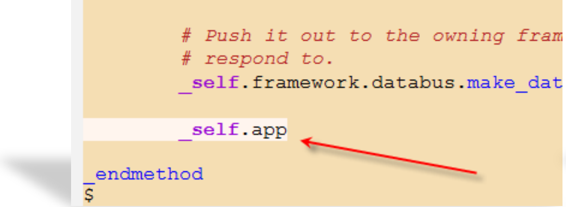
```
#% text_encoding = iso8859_1
_package sw
#
#MagikSF> object.define_method_target.metadata[:behaviour_sinks].p
```

mouse-1: switch to buffer "draw_arrow.magik"
mouse-2: pop to buffer, mouse-3: delete other windows

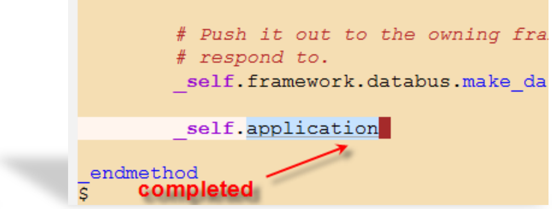
- Save/Restore editing Sessions, which allows you with one click of a button to load multiple files from a previous closed editing session. Even a cursor positions in the open files will be preserved from the previous session.



- Dynamic Abbreviation - it is standard function in the Emacs, but, from my experience, not everybody knows about. It is allowing you to write just a few characters of words you've written earlier to be able to expand them. Possible completions searched in the text in the current buffer, and if not found there, in all open buffers. To expand a word, just put the point right after the word and press <Alt-/>. As soon as you start typing – you are able to use this feature. Typing a next letter will automatically narrow your possible completions.

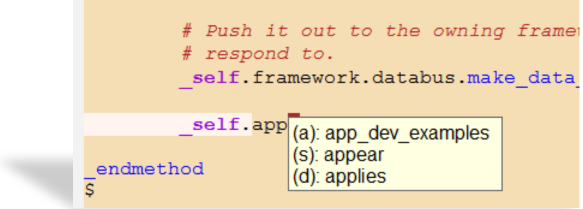


```
        # Push it out to the owning fram
        # respond to.
        _self.framework.databus.make_dat

        _self.app

_endmethod
$
```



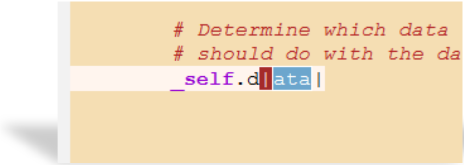
```
        # Push it out to the owning fra
        # respond to.
        _self.framework.databus.make_da

        _self.application

_endmethod
$
```

- Dynamic Abbreviation with Expansion of Multiple choices – press the same key second time and you will get a multi-choice candidates in a tooltip.

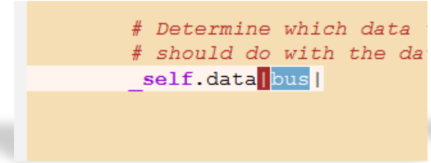


```
        # Push it out to the owning frame
        # respond to.
        _self.framework.databus.make_data_

        _self.app
                  (a): app_dev_examples
_endmethod        (s): appear
$                 (d): applies
```

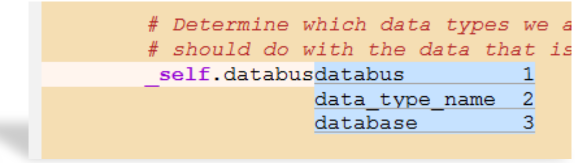
- Predictive Abbreviation. It's fairly similar to Dynamic Abbreviation expansion, which works based on the contents of the current buffer (or other buffers). Predictive abbreviation expansion works based on the previously written text. Unlike Dynamic Abbreviation, the text is analysed during idle time, while Emacs is doing nothing else. As you type you will be offered possible expansion for the word at the left from the cursor position. Typing new characters will automatically offer different completion. If the correct completion is offered – just press <Tab> to complete the word. Light blue colour indicates that there are possible different candidates for the expansion. To see other candidates – press <Tab> once again and choice of possible completions will be shown. If the completion is shown in the dark blue colour – that's mean this is the only possible candidate.



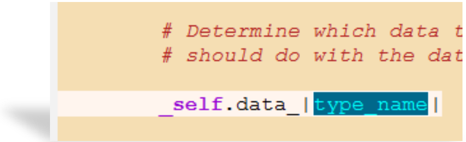
```
        # Determine which data
        # should do with the da
        _self.data
```



```
        # Determine which data
        # should do with the da
        _self.databus
```

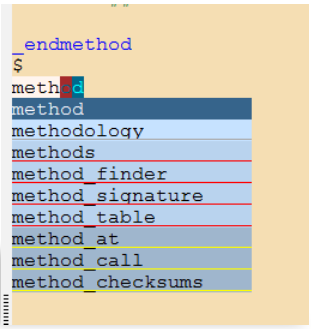


```
        # Determine which data types we a
        # should do with the data that is
        _self.databusdatabus          1
                     data_type_name   2
                     database         3
```

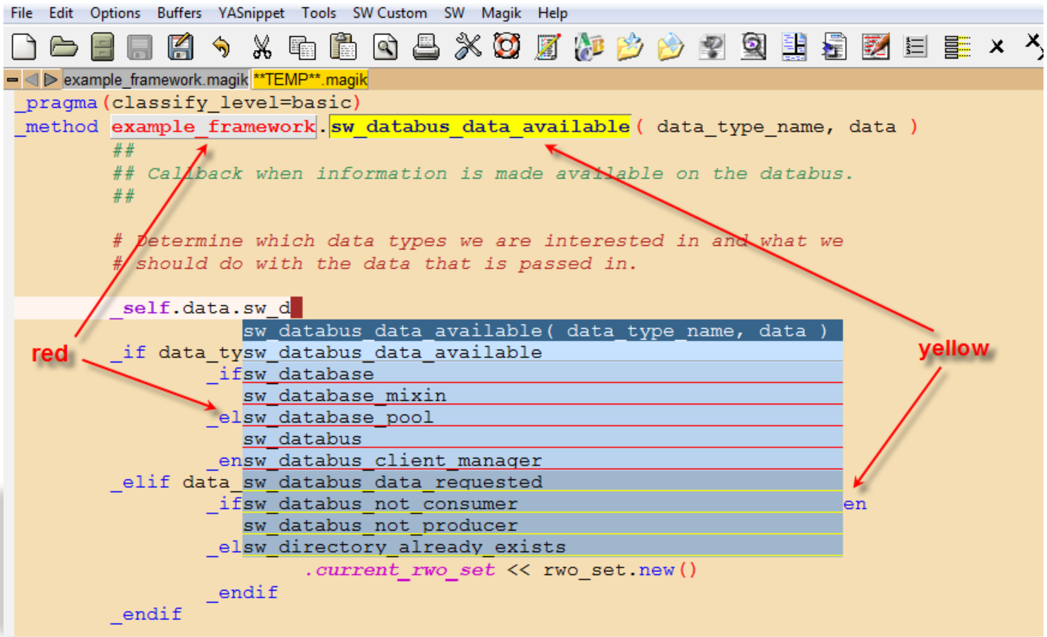


```
        # Determine which data t
        # should do with the dat

        _self.data_type_name
```

- Auto Complete - provides a way to complete any strings you expected by a graphical pull down menu. It looks like what Visual Studio provides as IntelliSense, Eclipse provides as CodeAssist. It is not configured to start automatically, you will need to use <Alt-[> to get Auto Complete menu and <Alt-]> to close it if you not finding suitable candidate. But it is possible to configure that Auto Complete will start automatically after you typed few characters. If this is enabled – then you will see the list of possible completions as you type. Use <Tab> to navigate through the list.



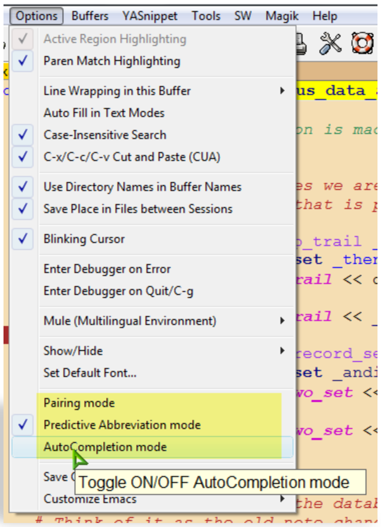
```
_endmethod
$
meth d
method
methodology
methods
method_finder
method_signature
method_table
method_at
method_call
method_checksums
```



```
_pragma(classify_level=basic)
_method example_framework.sw_databus_data_available( data_type_name, data )
        ##
        ## Callback when information is made available on the databus.
        ##

        # Determine which data types we are interested in and what we
        # should do with the data that is passed in.

        _self.data.sw_d
                  sw_databus_data_available( data_type_name, data )
        _if data_tysw_databus_data_available
                _ifsw_database
                  sw_database_mixin
                _elsw_database_pool
                  sw_databus
                _ensw_databus_client_manager
        _elif data_sw_databus_data_requested
                _ifsw_databus_not_consumer                              en
                  sw_databus_not_producer
                _elsw_directory_already_exists
                     .current_rwo_set << rwo_set.new()
                _endif
        _endif
```

Auto Complete unlike previous expansion tools working with the Method Finder. All possible candidates are shown from the Classes and Methods available in the current application image.   Candidates are shown in accordance with the code font locking – Classes fetched by the Method Finder underlined in red, Methods found by the Method Finder underlined in yellow, words found in open buffers shown without underlining.

I am planning to place ON/OFF toggles on the Toolbar for Auto Complete and Predictive Abbreviation modes in order for users to decide which mode is preferable.

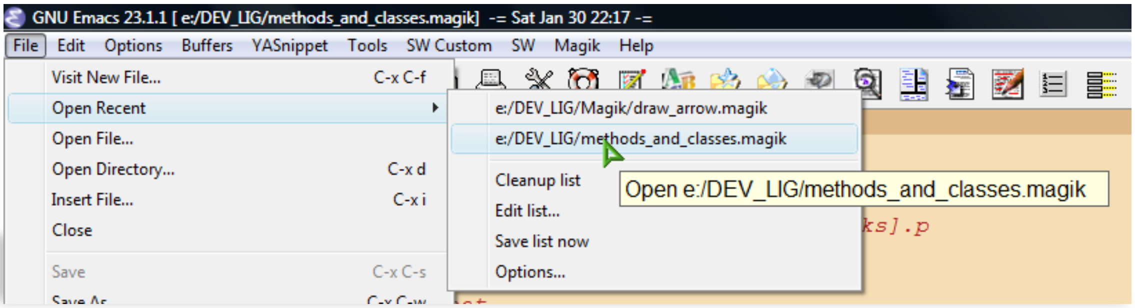

Options
Buffers
YASnippet
Tools
SW
Magik
Help
✓ Active Region Highlighting
✓ Paren Match Highlighting
Line Wrapping in this Buffer
Auto Fill in Text Modes
✓ Case-Insensitive Search
✓ C-x/C-c/C-v Cut and Paste (CUA)
✓ Use Directory Names in Buffer Names
✓ Save Place in Files between Sessions
✓ Blinking Cursor
Enter Debugger on Error
Enter Debugger on Quit/C-g
Mule (Multilingual Environment)
Show/Hide
Set Default Font...
Pairing mode
✓ Predictive Abbreviation mode
AutoCompletion mode
Save
Customize Emacs
Toggle ON/OFF AutoCompletion mode

- Automatic insertion of corresponding ({[and ". Thus typing `(' will normally insert `()' and put cursor between them. Otherwise, when the region is active, it will be wrapped in the parentheses.

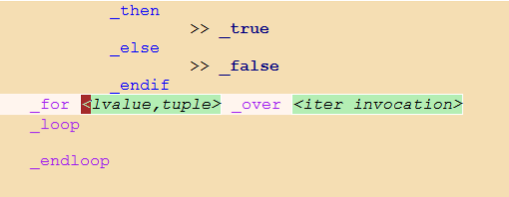
- Use Options menu to activate/deactivate those modes.

- Highlight text matching PATTERN. Examples: In a source code highlight a variable to quickly see all places it is modified or referenced. New occurrences of that variable will be highlighted as they are typed. In a *gis* or other buffer that is showing lots of output, highlight the parts of the output you're interested in. Those patterns will be highlighted as soon as they appear in the buffer. Useful if you have a lot of output during debugging. This option is located on the Edit menu.

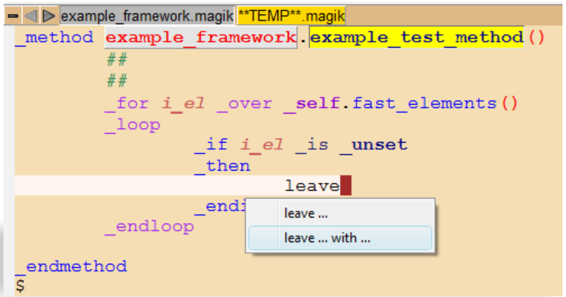
- Additional menu item for visiting files that were operated on recently


GNU Emacs 23.1.1 [e:/DEV_LIG/methods_and_classes.magik] -= Sat Jan 30 22:17 -=
File
Edit
Options
Buffers
YASnippet
Tools
SW Custom
SW
Magik
Help
Visit New File... C-x C-f
Open Recent
Open File...
Open Directory... C-x d
Insert File... C-x i
Close
Save C-x C-s
Save As C-x C-w
e:/DEV_LIG/Magik/draw_arrow.magik
e:/DEV_LIG/methods_and_classes.magik
Cleanup list
Edit list...
Save list now
Options...
Open e:/DEV_LIG/methods_and_classes.magik

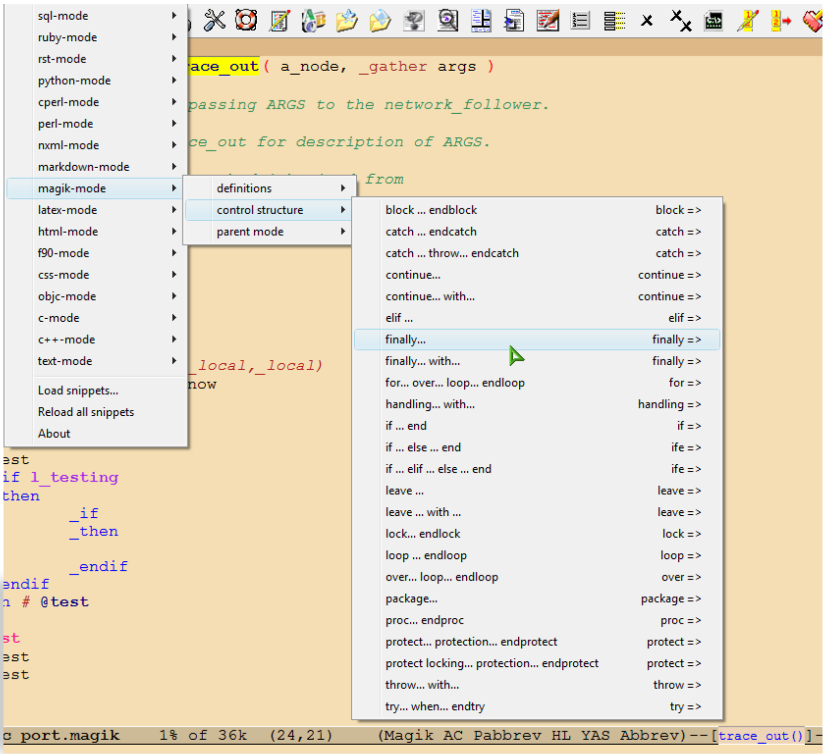
- Snippets for Magik. It allows you to type an abbreviation and automatically expand it into function templates (similar to electric-magik). Navigation between snippet fields – using <Tab>. Templates will expand automatically as you type them and they are available via menu too.



```
                _then
                        >> _true
                _else
                        >> _false
                _endif
_for <lvalue,tuple> _over <iter invocation>
_loop

_endloop
```

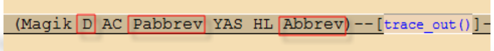


```
_method example_framework.example_test_method()
        ##
        ##
        _for i_el _over _self.fast_elements()
        _loop
                _if i_el _is _unset
                _then
                        leave
                _endif
        _endloop

_endmethod
$
```

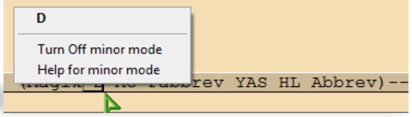

sql-mode
ruby-mode
rst-mode
python-mode
cperl-mode
perl-mode
nxml-mode
markdown-mode
magik-mode
latex-mode
html-mode
f90-mode
css-mode
objc-mode
c-mode
c++-mode
text-mode
Load snippets...
Reload all snippets
About
definitions
control structure
parent mode
block ... endblock block =>
catch ... endcatch catch =>
catch ... throw... endcatch catch =>
continue... continue =>
continue... with... continue =>
elif ... elif =>
finally... finally =>
finally... with... finally =>
for... over... loop... endloop for =>
handling... with... handling =>
if ... end if =>
if ... else ... end ife =>
if ... elif ... else ... end ife =>
leave ... leave =>
leave ... with ... leave =>
lock... endlock lock =>
loop ... endloop loop =>
over... loop... endloop over =>
package... package =>
proc... endproc proc =>
protect... protection... endprotect protect =>
protect locking... protection... endprotect protect =>
throw... with... throw =>
try... when... endtry try =>

```
ace_out( a_node, _gather args )
passing ARGS to the network_follower.
ce_out for description of ARGS.
                                from
_local,_local)
now
est
if l_testing
then
        _if
        _then

        _endif
endif
n # @test

st
est
est
c port.magik     1% of 36k   (24,21)     (Magik AC Pabbrev HL YAS Abbrev)--[trace_out()]-
```

- Dedicated mode - when a buffer is "dedicated" or locked, Emacs will not select files into that buffer. Dedicated buffers will have "D" shown in the mode line and marked lock icon. I found it very useful to lock the *gis* buffer. NOTE: All minor modes are shown in the mode line and you can use mouse to switch it OFF or get Help on the specific mode.

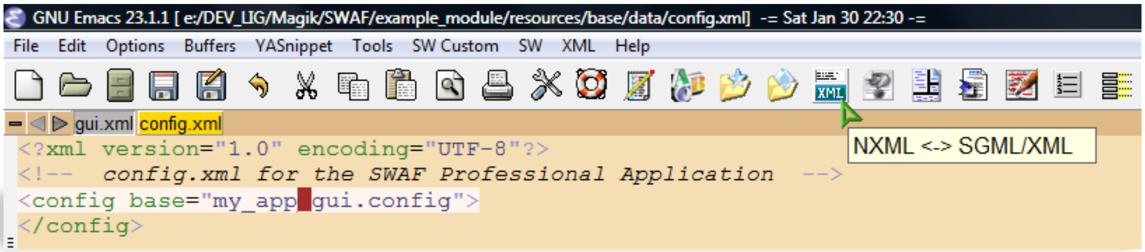


```
(Magik D AC Pabbrev YAS HL Abbrev)--[trace_out()]-
```



```
(Magik D AC Pabbrev YAS HL Abbrev)--
```

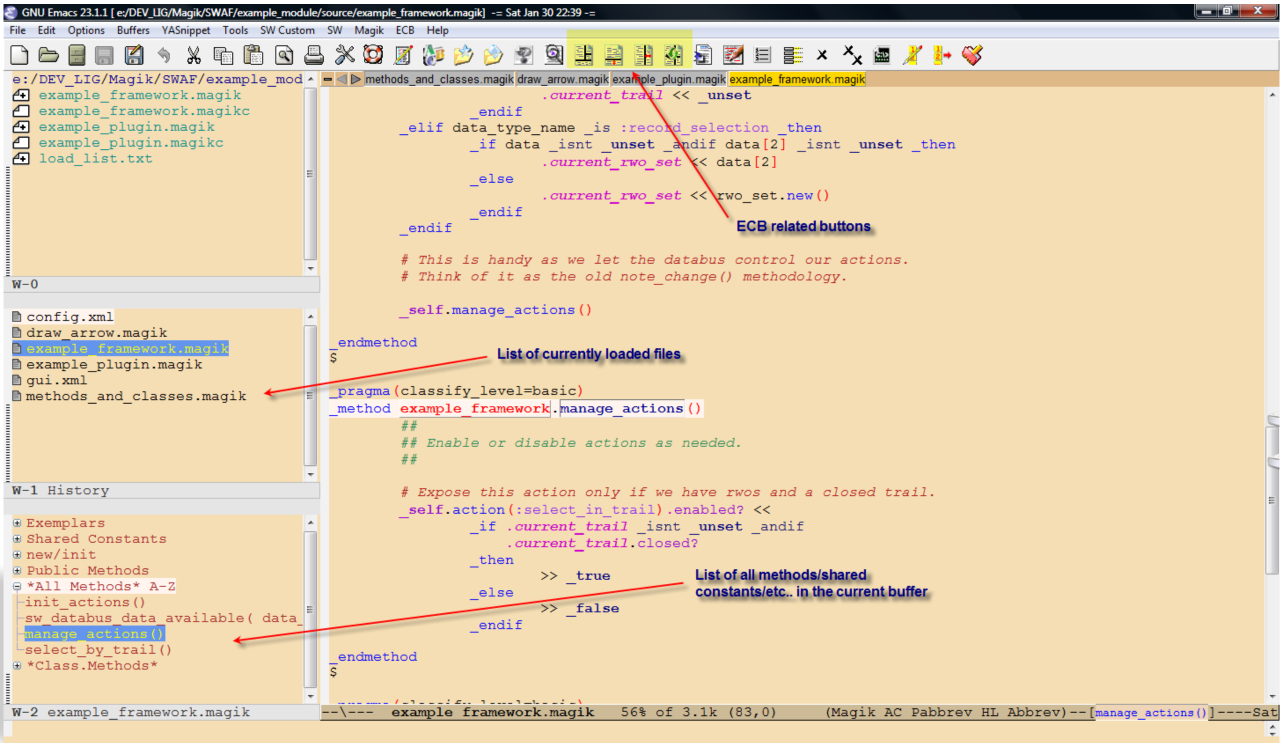
- NXML <-> SGML/XML mode switch


GNU Emacs 23.1.1 [e:/DEV_LIG/Magik/SWAF/example_module/resources/base/data/config.xml] -= Sat Jan 30 22:30 -=
File Edit Options Buffers YASnippet Tools SW Custom SW XML Help
gui.xml config.xml

NXML <-> SGML/XML

```
<?xml version="1.0" encoding="UTF-8"?>
<!--  config.xml for the SWAF Professional Application  -->
<config base="my_app gui.config">
</config>
```

- Class Browser – F3-j / F3-b – allows you to go back in the File Navigation history, so you are able to return to the point in a code where you came from during your F3-j key journey.
- F4-m / F12-m – copy current magik method to the Work (**Temp**.magik) buffer. Very helpful while making temporary changes in the code or just testing/debugging.
- F2-b – to compile message files <*.msg> directly from Emacs.
-
- ECB – Emacs Code Browser which displays a number of informational windows that allow for easy source code navigation and overview.


GNU Emacs 23.1.1 [e:/DEV_LIG/Magik/SWAF/example_module/source/example_framework.magik] -= Sat Jan 30 22:39 -=
File Edit Options Buffers YASnippet Tools SW Custom SW Magik ECB Help
methods_and_classes.magik draw_arrow.magik example_plugin.magik example_framework.magik

```
e:/DEV_LIG/Magik/SWAF/example_mod
  example_framework.magik
  example_framework.magikc
  example_plugin.magik
  example_plugin.magikc
  load_list.txt
W-0

config.xml
draw_arrow.magik
example_framework.magik
example_plugin.magik
gui.xml
methods_and_classes.magik
W-1 History

⊞ Exemplars
⊞ Shared Constants
⊞ new/init
⊞ Public Methods
⊟ *All Methods* A-Z
  ├init_actions()
  ├sw_databus_data_available( data_
  ├manage_actions()
  └select_by_trail()
⊞ *Class.Methods*
W-2 example_framework.magik
```

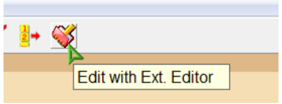
```
                                .current_trail << _unset
                        _endif
                _elif data_type_name _is :record_selection _then
                        _if data _isnt _unset _andif data[2] _isnt _unset _then
                                .current_rwo_set << data[2]
                        _else
                                .current_rwo_set << rwo_set.new()
                        _endif
                _endif

                # This is handy as we let the databus control our actions.
                # Think of it as the old note_change() methodology.

                _self.manage_actions()

_endmethod
$

_pragma(classify_level=basic)
_method example_framework.manage_actions()
        ##
        ## Enable or disable actions as needed.
        ##

        # Expose this action only if we have rwos and a closed trail.
        _self.action(:select_in_trail).enabled? <<
                _if .current_trail _isnt _unset _andif
                    .current_trail.closed?
                _then
                        >> _true
                _else
                        >> _false
                _endif

_endmethod
$
--\---  example framework.magik   56% of 3.1k  (83,0)     (Magik AC Pabbrev HL Abbrev)--[manage_actions()]----Sat
```

ECB related buttons

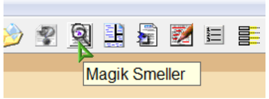
List of currently loaded files

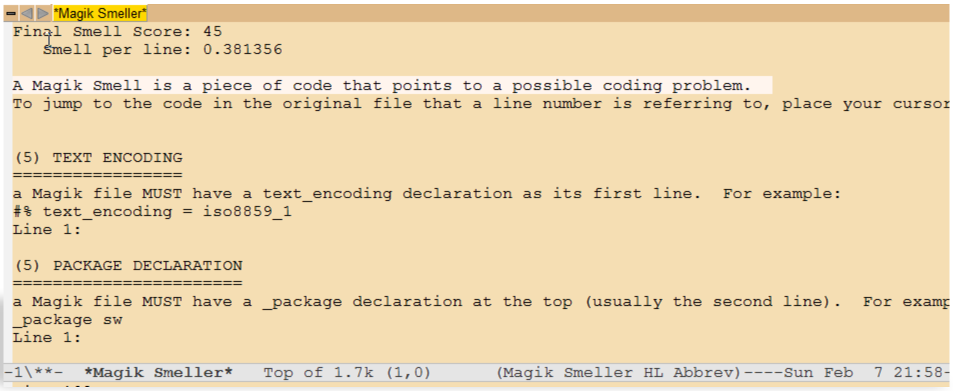
List of all methods/shared constants/etc.. in the current buffer

- Integration with an External Editor. This will allow you to load the file from the Emacs, if necessary, into an external editor of your choice to do some additional editing/searching or other file manipulations.


Edit with Ext. Editor

- Magik Smeller, which helps you to find possible problems in the code. A tool to interactively report actual or possible code syntax problems that are acceptable to the Magik compiler but do not meet development standards or may be considered inefficient.
  A new buffer called ***Magik Smeller*** will appear showing a list of all possible code "smells", their line number in the Magik file and a short description. To jump back to a line of code that has been reported as a smell, place your cursor on that line in the ***Magik Smeller*** and press "**F3-j**".

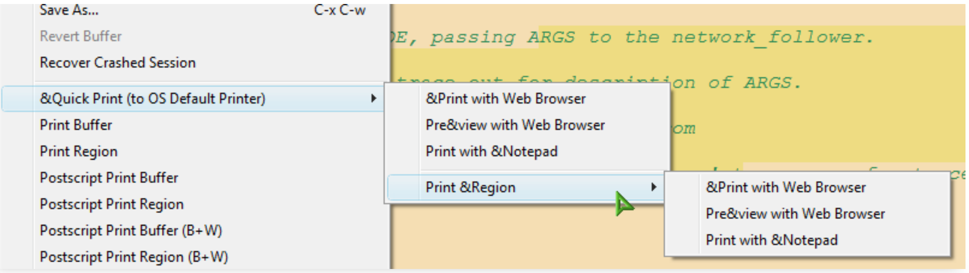




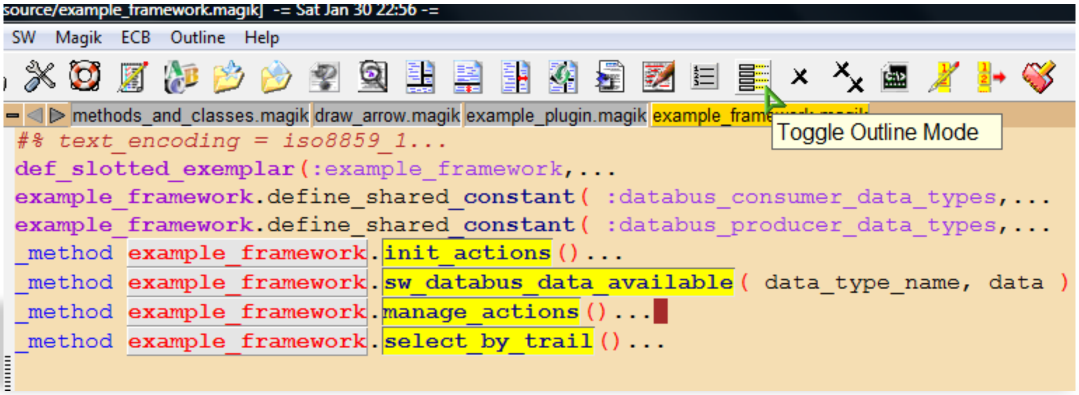
```
*Magik Smeller*
Final Smell Score: 45
   Smell per line: 0.381356

A Magik Smell is a piece of code that points to a possible coding problem.
To jump to the code in the original file that a line number is referring to, place your cursor


(5) TEXT ENCODING
=================
a Magik file MUST have a text_encoding declaration as its first line.  For example:
#% text_encoding = iso8859_1
Line 1:

(5) PACKAGE DECLARATION
=======================
a Magik file MUST have a _package declaration at the top (usually the second line).  For examp
_package sw
Line 1:

-1\**-  *Magik Smeller*   Top of 1.7k (1,0)      (Magik Smeller HL Abbrev)----Sun Feb  7 21:58-
```

-

- Print in colour or black&white directly from the Emacs. Preview in colour – will create HTML page with current fontlocking highlights in the code.

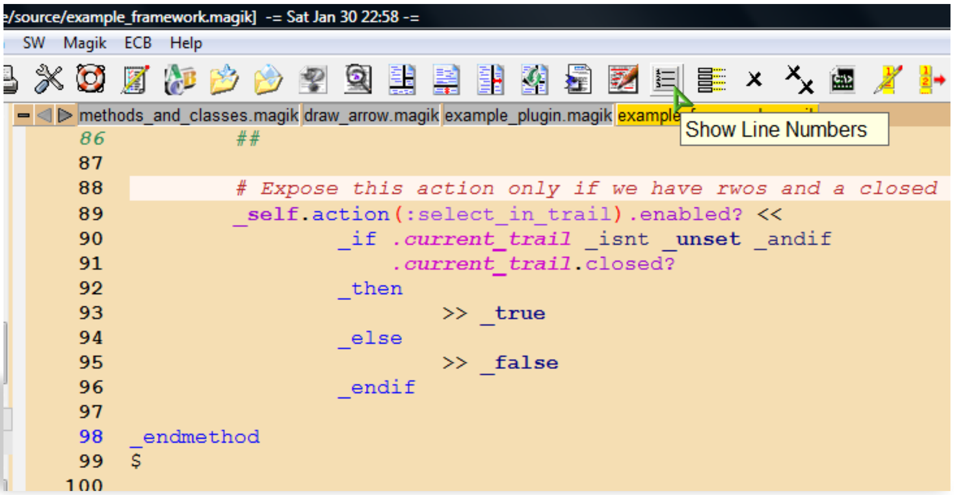


- Outline mode for Magik files. Using keys <Alt>+Up/Down in the Outline mode the Magik file can be collapsed to display only the method names and methods can be unfolded one by one for editing.


source/example_framework.magik] -= Sat Jan 30 22:56 -=
SW Magik ECB Outline Help
methods_and_classes.magik
draw_arrow.magik
example_plugin.magik
example_framework.magik
Toggle Outline Mode

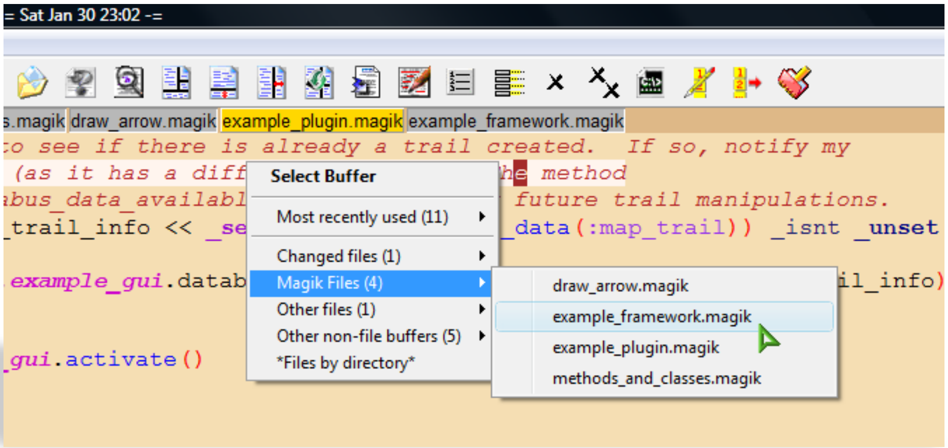
```
#% text_encoding = iso8859_1...
def_slotted_exemplar(:example_framework,...
example_framework.define_shared_constant( :databus_consumer_data_types,...
example_framework.define_shared_constant( :databus_producer_data_types,...
_method example_framework.init_actions()...
_method example_framework.sw_databus_data_available( data_type_name, data )
_method example_framework.manage_actions()...
_method example_framework.select_by_trail()...
```

- Show/hide Line numbers in a buffer.


e/source/example_framework.magik] -= Sat Jan 30 22:58 -=
SW
Magik
ECB
Help
methods_and_classes.magik
draw_arrow.magik
example_plugin.magik
example
Show Line Numbers

```
            ##

            # Expose this action only if we have rwos and a closed
            _self.action(:select_in_trail).enabled? <<
                    _if .current_trail _isnt _unset _andif
                        .current_trail.closed?
                    _then
                            >> _true
                    _else
                            >> _false
                    _endif

_endmethod
$
```

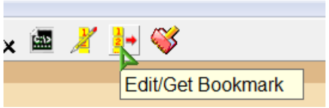
- Buffer selection- <Ctrl>+<left mouse button> will bring up the floating menu at the cursor position OR use <Ctrl>+<Tab> to navigate between tabs in the Tab-bar.



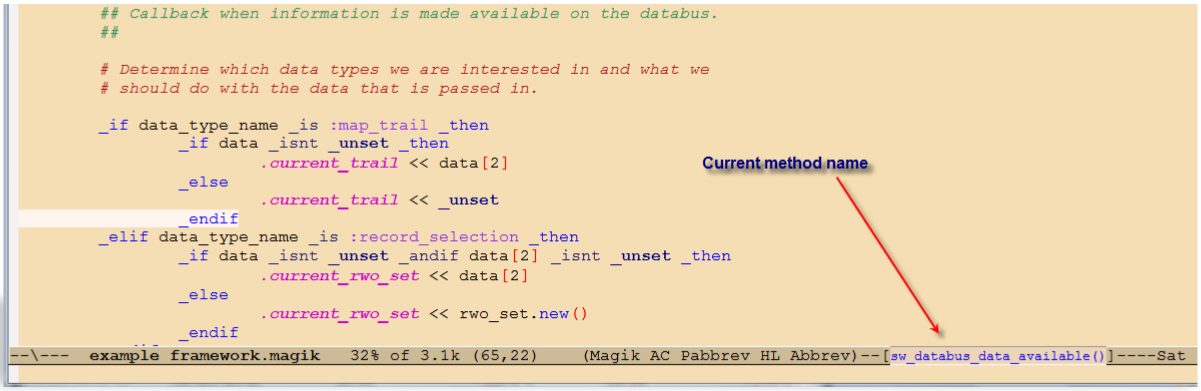
```
to see if there is already a trail created.  If so, notify my
 (as it has a diff                  he method
abus_data_availabl                 future trail manipulations.
trail_info << _se                  _data(:map_trail)) _isnt _unset

.example_gui.datab                                          il_info)


gui.activate()
```

- Save all file backups (*.magik~) in the single configurable directory (system %TEMP% – by default).

- Use icons to work with bookmarks OR use ‘breadcrumbs’ <Esc-Spc> or <Alt-Spc> - a series of breadcrumb bookmarks with quick cycling though them using <F5-b>. Bookmarks are persistent across Emacs editing sessions.

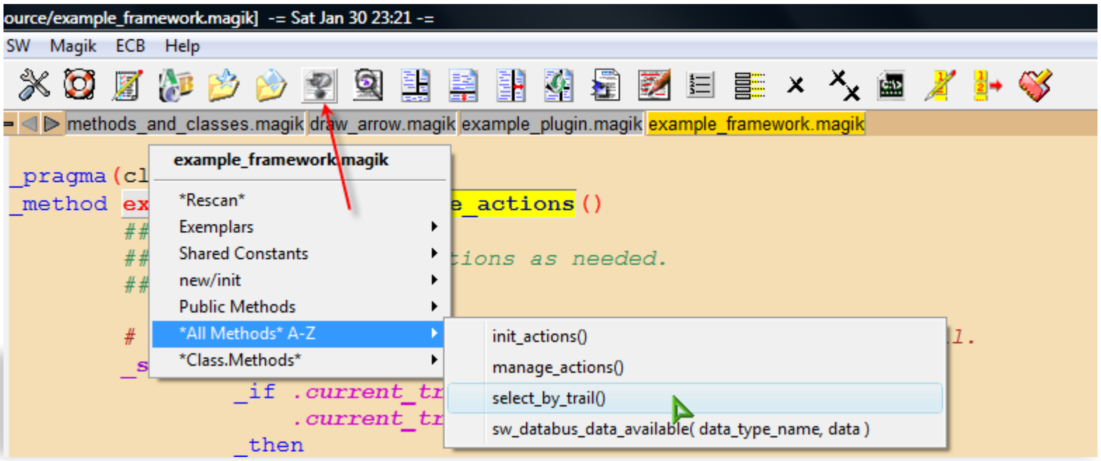


- Method name in the mode line always visible

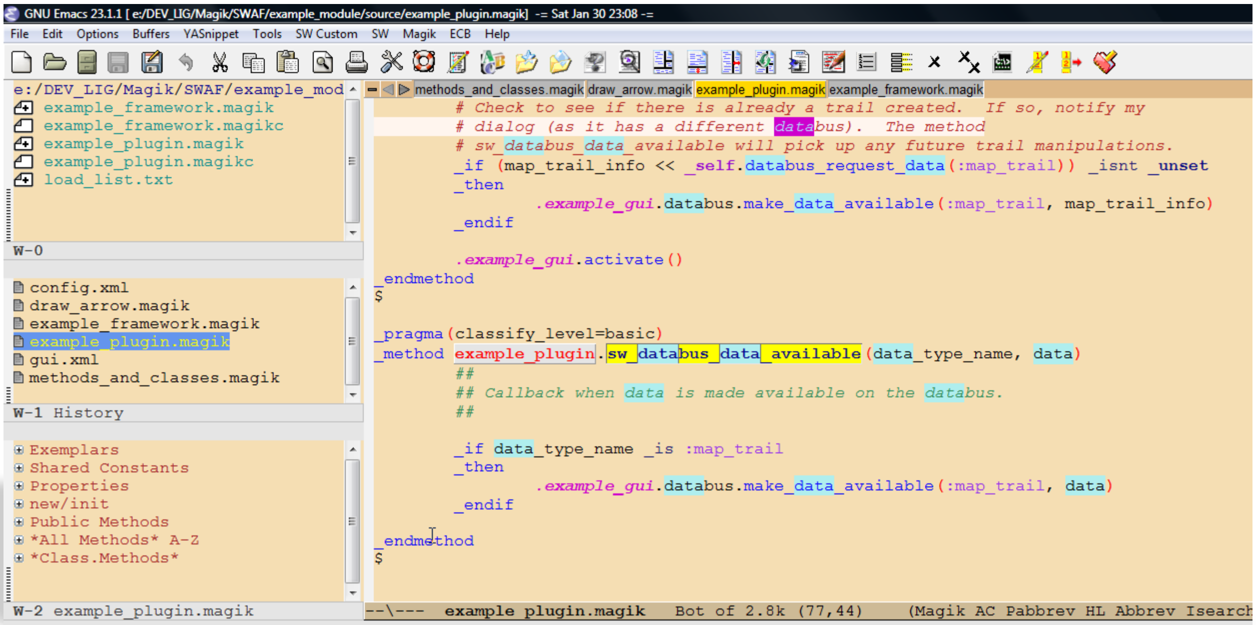


```
        ## Callback when information is made available on the databus.
        ##

        # Determine which data types we are interested in and what we
        # should do with the data that is passed in.

        _if data_type_name _is :map_trail _then
                _if data _isnt _unset _then
                        .current_trail << data[2]
                _else
                        .current_trail << _unset
                _endif
        _elif data_type_name _is :record_selection _then
                _if data _isnt _unset _andif data[2] _isnt _unset _then
                        .current_rwo_set << data[2]
                _else
                        .current_rwo_set << rwo_set.new()
                _endif
--\---  example framework.magik   32% of 3.1k (65,22)     (Magik AC Pabbrev HL Abbrev)--[sw_databus_data_available()]----Sat
```

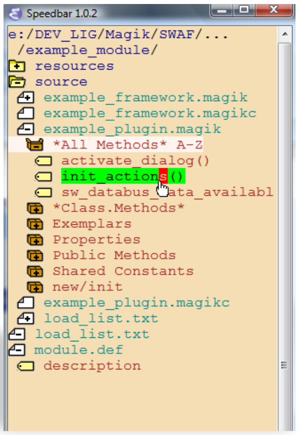
- Show Methods menu allowing quickly navigate between methods.



- Additional highlighting during the incremental search



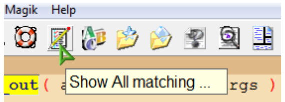
```
        # Check to see if there is already a trail created.  If so, notify my
        # dialog (as it has a different databus).  The method
        # sw_databus_data_available will pick up any future trail manipulations.
        _if (map_trail_info << _self.databus_request_data(:map_trail)) _isnt _unset
        _then
                .example_gui.databus.make_data_available(:map_trail, map_trail_info)
        _endif

        .example_gui.activate()
_endmethod
$

_pragma(classify_level=basic)
_method example_plugin.sw_databus_data_available(data_type_name, data)
        ##
        ## Callback when data is made available on the databus.
        ##

        _if data_type_name _is :map_trail
        _then
                .example_gui.databus.make_data_available(:map_trail, data)
        _endif

_endmethod
$
--\---  example plugin.magik   Bot of 2.8k (77,44)     (Magik AC Pabbrev HL Abbrev Isearch
```

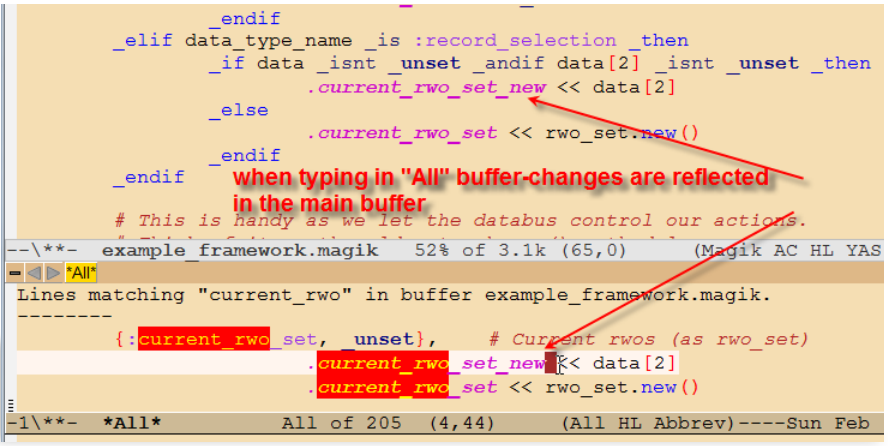


```
e:/DEV_LIG/Magik/SWAF/...
 /example_module/
[+] resources
[-] source
 [+] example_framework.magik
 [ ] example_framework.magikc
 [-] example_plugin.magik
   *All Methods* A-Z
    activate_dialog()
    init_actions()
    sw_databus_data_availabl
   *Class.Methods*
   Exemplars
   Properties
   Public Methods
   Shared Constants
   new/init
 [ ] example_plugin.magikc
 [+] load_list.txt
[-] load_list.txt
[-] module.def
 description
```

- Speedbar (F11) - a special skinny frame with a specialized directory listing in it. This listing will have both directories and filtered files in it. You can then load files into your Emacs buffer, or expand the files to display all the methods/progs/constants/…. in them and jump to those tags. You can also expand multiple directories into your Speedbar frame.

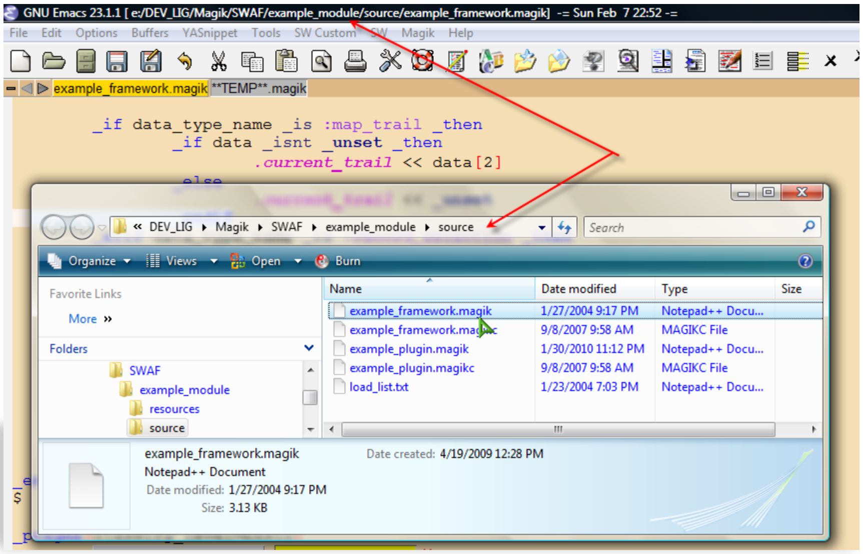
- Use 'Alt-x all <Enter>' or the icon to list\EDIT all lines matching a given regexp in *All* buffer. ***Changes made in the *All* buffer is propagated to the original buffer.*** Use 'F3-j' from *All* buffer to go to a match in the original file. **Note that changes to the `*All*' buffer are propagated back to the original buffer.**




when typing in "All" buffer-changes are reflected in the main buffer
All

```
            _endif
      _elif data_type_name _is :record_selection _then
            _if data _isnt _unset _andif data[2] _isnt _unset _then
                  .current_rwo_set_new << data[2]
            _else
                  .current_rwo_set << rwo_set.new()
            _endif
      _endif
      # This is handy as we let the databus control our actions.
--\**-  example_framework.magik   52% of 3.1k (65,0)      (Magik AC HL YAS
Lines matching "current_rwo" in buffer example_framework.magik.
--------
      {:current_rwo_set, _unset},      # Current rwos (as rwo_set)
                  .current_rwo_set_new << data[2]
                  .current_rwo_set << rwo_set.new()
-1\**-  *All*        All of 205  (4,44)      (All HL Abbrev)----Sun Feb
```

- While you have a file loaded in to the Emacs buffer pressing <F12-F12> will open Windows Explorer in the directory of the current file.

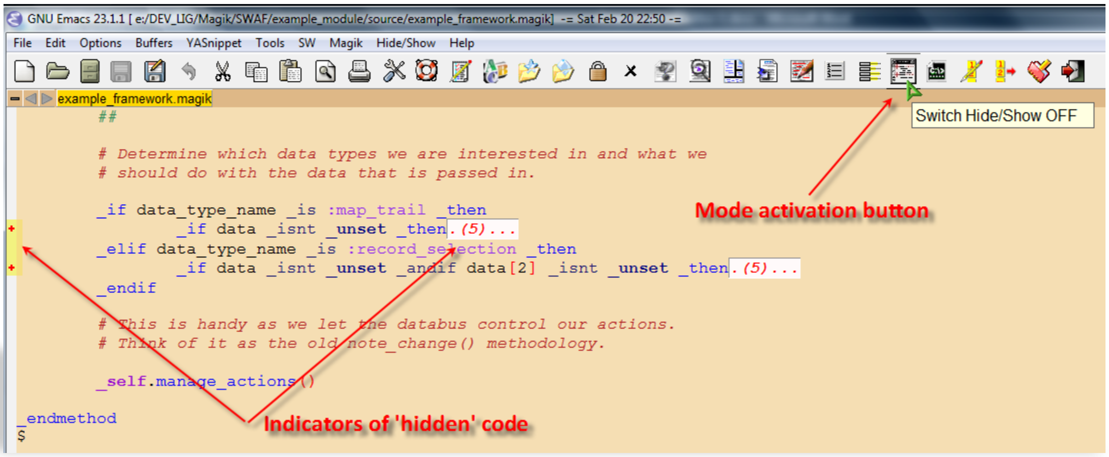


```
      _if data_type_name _is :map_trail _then
            _if data _isnt _unset _then
                  .current_trail << data[2]
```

Name	Date modified	Type	Size
example_framework.magik	1/27/2004 9:17 PM	Notepad++ Docu...	
example_framework.magikc	9/8/2007 9:58 AM	MAGIKC File	
example_plugin.magik	1/30/2010 11:12 PM	Notepad++ Docu...	
example_plugin.magikc	9/8/2007 9:58 AM	MAGIKC File	
load_list.txt	1/23/2004 7:03 PM	Notepad++ Docu...	

example_framework.magik
Notepad++ Document
Date modified: 1/27/2004 9:17 PM
Size: 3.13 KB
Date created: 4/19/2009 12:28 PM

- Magik code folding – where large blocks of code can be distracting. With the code folding feature you can hide the code not important to your current work. Code folding cleans your work space. Use <Ctrl-.> key to hide the block of code while your cursor is inside the desired block.



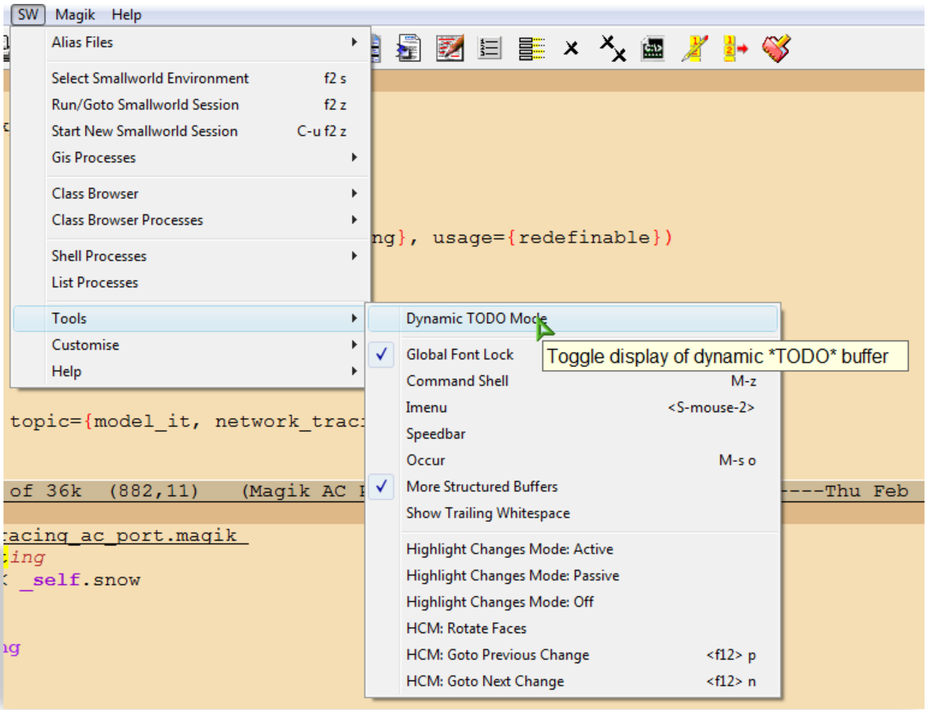
GNU Emacs 23.1.1 [e:/DEV_LIG/Magik/SWAF/example_module/source/example_framework.magik] -= Sat Feb 20 22:50 -=
File Edit Options Buffers YASnippet Tools SW Magik Hide/Show Help
example_framework.magik
Switch Hide/Show OFF

```
        ##

        # Determine which data types we are interested in and what we
        # should do with the data that is passed in.

        _if data_type_name _is :map_trail _then
                _if data _isnt _unset _then. (5)...
        _elif data_type_name _is :record_selection _then
                _if data _isnt _unset _andif data[2] _isnt _unset _then. (5)...
        _endif

        # This is handy as we let the databus control our actions.
        # Think of it as the old note_change() methodology.

        _self.manage_actions()

_endmethod
$
```

Mode activation button

Indicators of 'hidden' code

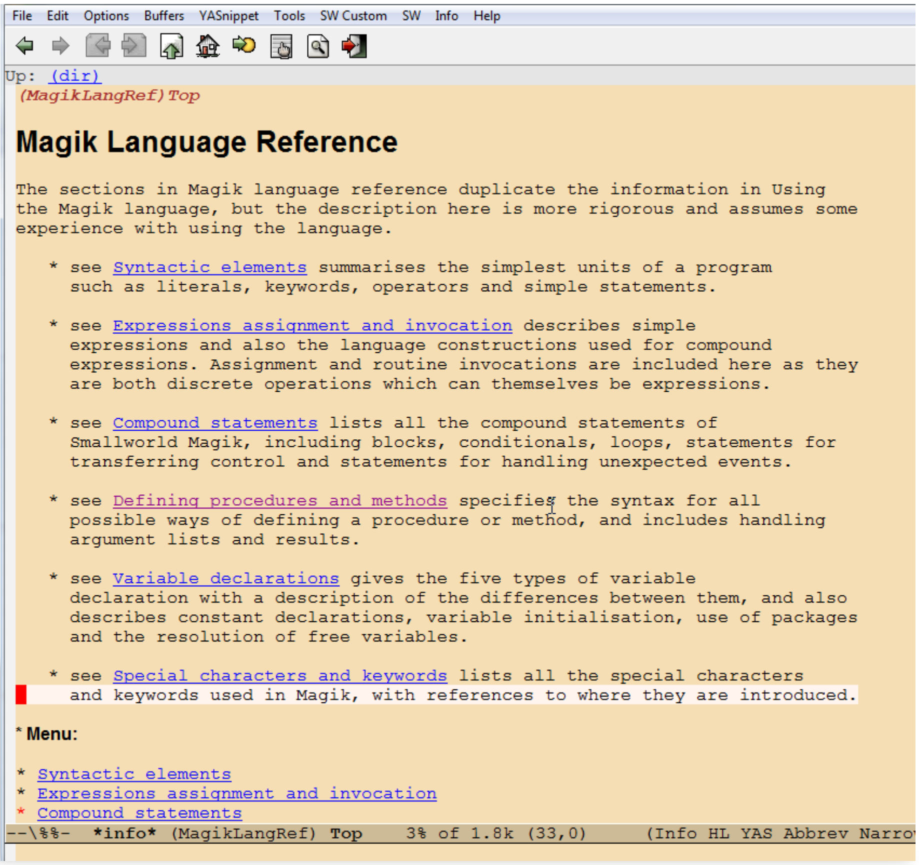
- Use set of very useful Tools available on the SW menu.



SW Magik Help
Alias Files
Select Smallworld Environment f2 s
Run/Goto Smallworld Session f2 z
Start New Smallworld Session C-u f2 z
Gis Processes
Class Browser
Class Browser Processes
Shell Processes
List Processes
Tools
Customise
Help
Dynamic TODO Mode
✓ Global Font Lock
Command Shell M-z
Imenu <S-mouse-2>
Speedbar
Occur M-s o
✓ More Structured Buffers
Show Trailing Whitespace
Highlight Changes Mode: Active
Highlight Changes Mode: Passive
Highlight Changes Mode: Off
HCM: Rotate Faces
HCM: Goto Previous Change <f12> p
HCM: Goto Next Change <f12> n
Toggle display of dynamic *TODO* buffer

```
ng}, usage={redefinable})
topic={model_it, network_trac:
of 36k  (882,11)   (Magik AC I
----Thu Feb
racing_ac_port.magik
:ing
< _self.snow
ng
```

- Magik Language Reference available within Emacs.



Up: (dir)
(MagikLangRef) Top

Magik Language Reference

```
The sections in Magik language reference duplicate the information in Using
the Magik language, but the description here is more rigorous and assumes some
experience with using the language.

   * see Syntactic elements summarises the simplest units of a program
     such as literals, keywords, operators and simple statements.

   * see Expressions assignment and invocation describes simple
     expressions and also the language constructions used for compound
     expressions. Assignment and routine invocations are included here as they
     are both discrete operations which can themselves be expressions.

   * see Compound statements lists all the compound statements of
     Smallworld Magik, including blocks, conditionals, loops, statements for
     transferring control and statements for handling unexpected events.

   * see Defining procedures and methods specifies the syntax for all
     possible ways of defining a procedure or method, and includes handling
     argument lists and results.

   * see Variable declarations gives the five types of variable
     declaration with a description of the differences between them, and also
     describes constant declarations, variable initialisation, use of packages
     and the resolution of free variables.

   * see Special characters and keywords lists all the special characters
     and keywords used in Magik, with references to where they are introduced.
```

* Menu:

```
* Syntactic elements
* Expressions assignment and invocation
* Compound statements
--\%%-  *info* (MagikLangRef) Top    3% of 1.8k (33,0)      (Info HL YAS Abbrev Narro
```

- Use %USER_EMACS% environment variable in gis_aliases to load your local Emacs settings, like

  (setq programmer-tag "User Name, Company Name")

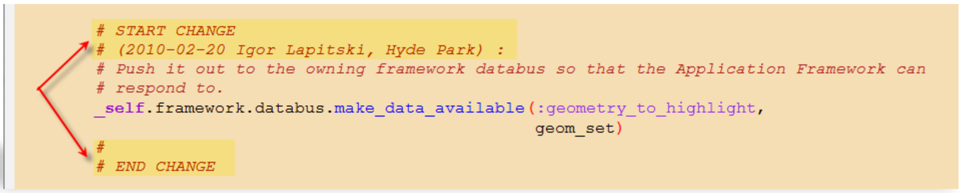
  (setq project-name "Project Name")

  (setq programmer-time-stamp-format "%:y-%02m-%02d"); %02H:%02M:%02S")

- [Alt-n] to insert-programmer-note;

  [Alt-m] to insert-programmer-modified-note;

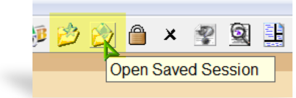
  [Alt-c] to insert-change-note around selected block – just select the block of code and press <Alt-c> and note change comments will be placed around the selected code; OR just use this to create a note change.



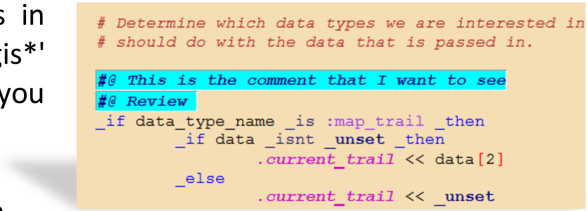
```
# START CHANGE
# (2010-02-20 Igor Lapitski, Hyde Park) :
# Push it out to the owning framework databus so that the Application Framework can
# respond to.
_self.framework.databus.make_data_available(:geometry_to_highlight,
                                            geom_set)

#
# END CHANGE
```

[Alt-h] to insert-method-comment;

- Save/Restore Emacs Sessions. That allows you to organize your different projects into "sessions". Each Session has a name (file name). When you 'Save Session', the list of all opened files is saved into the current session and all the files you were editing will be reloaded the next time you 'Restore Session'. This allows you to keep different files open in different Emacs sessions. By using 'Save/Restore Session' you are activating 'Session auto-saving mechanism' - on Emacs exit current Session will be saved automatically.



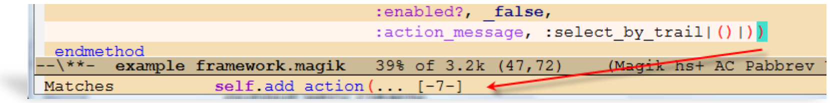
- To make prominent marks in '*gis*' buffer or comments in 'Magik' buffer use '@' at the beginning of the line in '*gis*' buffer and '#@' in 'Magik' buffer. I found this useful when you have a lots of output in the *gis* buffer and you need to see specific line – just use this in your code *write(“@Important info: “, a_variable)* and you will be able to see this line in the output very clearly.



```
# Determine which data types we are interested in
# should do with the data that is passed in.

#@ This is the comment that I want to see
#@ Review
_if data_type_name _is :map_trail _then
        _if data _isnt _unset _then
                .current_trail << data[2]
        _else
                .current_trail << _unset
```

- Use F2-b to send <*.msg> file to running GIS buffer. This will load current *.msg file into an open currently image.

- Emacs will display highlighting on whatever parenthesis matches the one before or after point. See message in Minibuffer describing the match when the matching parenthesis is off-screen. Message contains the linenumber.



```
                                      :enabled?, _false,
                                      :action_message, :select_by_trail|()|))
 endmethod
--\**-  example framework.magik   39% of 3.2k (47,72)     (Magik hs+ AC Pabbrev
Matches              self.add action(... [-7-]
```

- Use 'Ctrl-TAB' to switch\cycling between Emacs buffers.

- When using <Ctrl-x/Ctrl-b> to bring up Buffer Selection Menu – use following keys : <c> - to change the configuration; <s> - to save buffers marked with ’*’ ; <d> - to delete buffer

- List of key-bindings which might be useful in everyday Emacs operations with Smallworld environment:

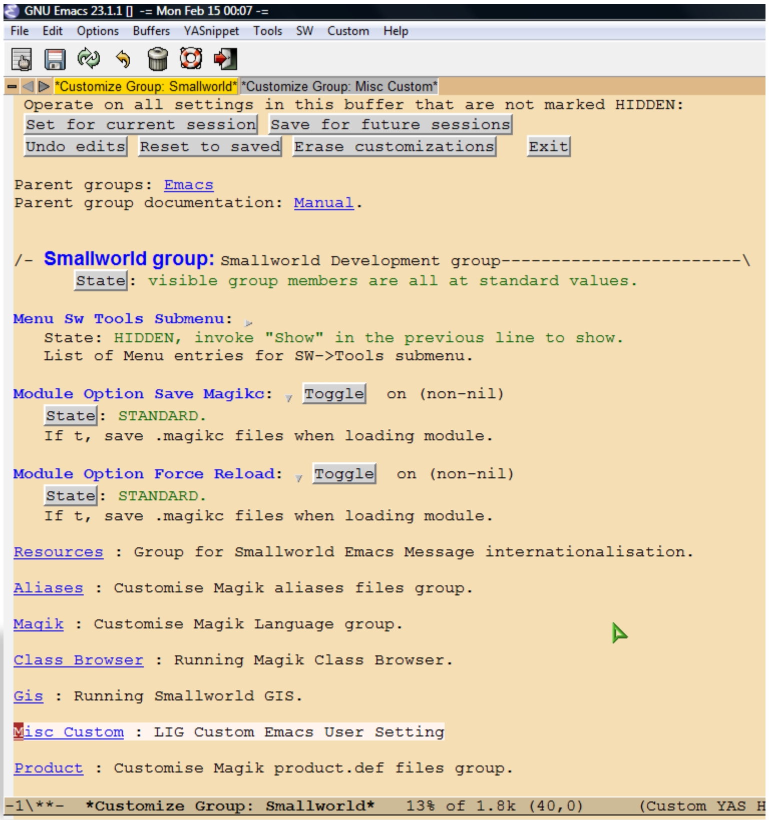
<f2> RET	magik-transmit-thing	<f3> /	cb-and-clear
<f2> SPC	explicit-electric-magik-space	<f3> ?	cb-help
<f2> #	magik-comment-region	<f3> b	cb-jump-back
<f2> $	magik-transmit-$-chunk	<f3> c	cb-paste-class
<f2> X	dev-tools-object-inspector	<f3> d	toggle-transmit-magik-debug-p
<f2> b	magik-transmit-buffer	<f3> j	cb-jump-to-source
<f2> h	magik-heading	<f3> m	cb-paste-method
<f2> m	magik-transmit-method	<f3> <f3>	cb
<f2> q	fill-magik-public-comment		
<f2> r	magik-transmit-region	<f4>	magik-f4-map
<f2> t	magik-trace-curr-statement	<f4> c	magik-copy-method
<f2> x	deep-print	<f4> d	dev-tools-vsd-transmit-method
<f2> <down>	forward-method	<f4> e	magik-ediff-methods

<f2> <f7>	magik-transmit-method	<f4> m	magik-copy-method-to-buffer
<f2> <f8>	magik-transmit-region	<f4> n	magik-set-work-buffer-name
<f2> <up>	backward-method	<f4> o	dev-tools-object-inspector
<f2> M-#	magik-uncomment-region	<f4> r	magik-copy-region-to-buffer
<C-f2>	magik-translate-old-vec-notation	<f4> s	magik-add-debug-statement
<f2> TAB	hippie-expand	<f4> w	magik-compare-methods
<f2> e	electric-magik-mode	<f4> <f4>	magik-symbol-complete
<f2> k	sw-reload-dotemacs	<f4> b	bury-buffer
<f2> s	gis-version-selection	<f4> g	goto-line
<f2> z	gis	<f6>	magik-copy-method
<f2> g	gis	<f7>	magik-transmit-method
		<f8>	magik-transmit-region
		<f9>	magik-mark-method
<M-Spc>	set breadcrumb bookmark	<f11>	speedbar-get-focus
<M-f5>	breadcrumb-list	<f12> -	ecb-cycle-maximized-ecb-buffers
<f5> b	breadcrumb-next	<f12> =	ecb-maximize-window-methods
<f5> d	bookmark-delete	<f12> b	ecb-minor-mode
<f5> e	edit-bookmarks	<f12> c	ecb-toggle-compile-window
<f5> j	edit-bookmarks	<f12> e	send-file-to-external-editor
<f5> l	bookmark-load	<f12> h	ecb-toggle-compile-window-height
<f5> m	bookmark-set	<f12> q	ecb-redraw-layout
<f5> s	bookmark-write	<f12> r	ecb-restore-window-sizes
<f12> x	switch-xml-mode	<f12> t	ecb-toggle-layout
<f12> <f12>	w32shell-explorer-here	<f12> u	ecb-rebuild-methods-buffer
<f12> m	magik-copy-method-to-buffer	<f12> w	ecb-toggle-ecb-windows
<f12> n	highlight-changes-next-change		
<f12> p	highlight-changes-previous-change		
<M-/>	dabbrev-expand	<M-S-down>	buf-move-down
<M-d>	close window	<M-S-left>	buf-move-left
<M-b>	move cursor to another window	<M-S-right>	buf-move-right
<M-f>	move cursor to another frame	<M-S-up>	buf-move-up
<M-m>	insert-programmer-modified-note	<M-c>	insert-change-note
<M-n>	insert-programmer-note	<M-h>	insert-method-comment
		<C-Tab>	toggle between files

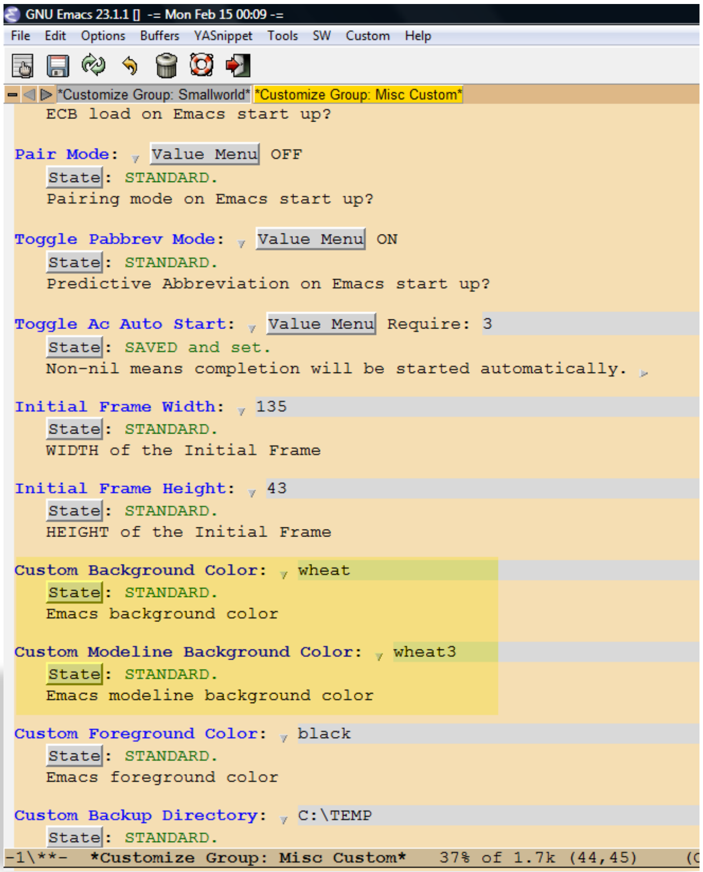
- For customisation of the empty magik file header and method comment – make necessary changes in the <~\site-lisp\custom-lisp\custom-magik-info.el> file.

Many of those settings are possible to customise, use the menu SW->Customose->Customise Smallworld Emacs.

The background colour of the custom version of the Emacs (wheat) also possible to change. All custom settings will be saved in the <.emacs-custom> file in your %HOME% directory. See http://www.hydepark-consulting.com/Emacs/ *Emacs_colours.pdf* for all possible colours.


GNU Emacs 23.1.1 [] -= Mon Feb 15 00:07 -=
File Edit Options Buffers YASnippet Tools SW Custom Help
Customize Group: Smallworld *Customize Group: Misc Custom*

```
 Operate on all settings in this buffer that are not marked HIDDEN:
 Set for current session  Save for future sessions
 Undo edits  Reset to saved  Erase customizations   Exit

Parent groups: Emacs
Parent group documentation: Manual.


/- Smallworld group: Smallworld Development group-----------------------\
      State: visible group members are all at standard values.

Menu Sw Tools Submenu:
   State: HIDDEN, invoke "Show" in the previous line to show.
   List of Menu entries for SW->Tools submenu.

Module Option Save Magikc:  Toggle  on (non-nil)
   State: STANDARD.
   If t, save .magikc files when loading module.

Module Option Force Reload:  Toggle  on (non-nil)
   State: STANDARD.
   If t, save .magikc files when loading module.

Resources : Group for Smallworld Emacs Message internationalisation.

Aliases : Customise Magik aliases files group.

Magik : Customise Magik Language group.

Class Browser : Running Magik Class Browser.

Gis : Running Smallworld GIS.

Misc Custom : LIG Custom Emacs User Setting

Product : Customise Magik product.def files group.

-1\**-  *Customize Group: Smallworld*   13% of 1.8k (40,0)     (Custom YAS H
```


GNU Emacs 23.1.1 [] -= Mon Feb 15 00:09 -=
File Edit Options Buffers YASnippet Tools SW Custom Help
Customize Group: Smallworld *Customize Group: Misc Custom*

```
   ECB load on Emacs start up?

Pair Mode:  Value Menu OFF
   State: STANDARD.
   Pairing mode on Emacs start up?

Toggle Pabbrev Mode:  Value Menu ON
   State: STANDARD.
   Predictive Abbreviation on Emacs start up?

Toggle Ac Auto Start:  Value Menu Require: 3
   State: SAVED and set.
   Non-nil means completion will be started automatically.

Initial Frame Width:  135
   State: STANDARD.
   WIDTH of the Initial Frame

Initial Frame Height:  43
   State: STANDARD.
   HEIGHT of the Initial Frame

Custom Background Color:  wheat
   State: STANDARD.
   Emacs background color

Custom Modeline Background Color:  wheat3
   State: STANDARD.
   Emacs modeline background color

Custom Foreground Color:  black
   State: STANDARD.
   Emacs foreground color

Custom Backup Directory:  C:\TEMP
   State: STANDARD.
-1\**-  *Customize Group: Misc Custom*   37% of 1.7k (44,45)    (C
```