






## Emacs support for REXX

Description	Keystroke	Function	Note
REXX Programming Language Support	Support for the <a href="#">REXX programming language</a> is minimal: you can activate the rexx-mode package by setting the <a href="#">pel-use-rexx</a> user option. <ul style="list-style-type: none"> <li>Files with the .rexx, .elx, .ncomm and .cpr are recognized as REXX source files and will automatically activate forth-mode if the package has been activated via that user-option.</li> <li>Generic programming language features like template text insertion handle REXX comment style. See <a href="#">🔗 Inserting Text</a> .</li> <li> REXX support is provided by <a href="#">rexx-mode</a> external package  automatically downloaded and installed by PEL when the <a href="#">pel-use-rexx</a> user option is set to <a href="#">t</a>.               <ul style="list-style-type: none"> <li>PEL adds the following regular expression to <a href="#">auto-mode-alist</a> to associate file extensions to the rexx-mode: “\\.\(rexx?\ elx\ ncomm\ cpr\)\ ” to support the following file extensions: .rex, .rexx, .elx, .ncomm and .cpr.</li> </ul> </li> <li> See also NetRexx support in <a href="#">🔗 I - NetRexx</a> provided by the the <a href="#">netrexx-mode.el</a> external package.</li> </ul>		
Open this PDF file. See also: <a href="#">🔗 Help/Info</a>	<div>&lt;f11&gt; SPC R &lt;f1&gt;</div> <div>&lt;f12&gt; &lt;f1&gt;</div>	(pel-help-pdf &optional OPEN-WEB-PAGE)	Open the <a href="#">🔗 I - REXX</a> local PDF. If the prefix argument (like <b>C-u</b> or <b>M--</b> ) is used, then it opens the remote GitHub hosted raw PDF instead. If the <a href="#">pel-flip-help-pdf-arg</a> user-option is set it's the other way around.
<a href="#">🔗 Customize</a> PEL REXX support	<div>&lt;f11&gt; SPC R &lt;f2&gt;</div> <div>&lt;f12&gt; &lt;f2&gt;</div>	(pel-customize-pel &optional OTHER-WINDOW)	Customize PEL REXX support. Use this to activate REXX support. <ul style="list-style-type: none"> <li>If OTHER-WINDOW is non-nil (use <b>C-u</b>), display in another window.</li> </ul>
<a href="#">🔗 Customize</a> Emacs REXX support	<div>&lt;f11&gt; SPC R &lt;f3&gt;</div> <div>&lt;f12&gt; &lt;f3&gt;</div>	(pel-customize-library &optional OTHER-WINDOW)	Customize Emacs REXX support: rexx-mode <ul style="list-style-type: none"> <li>If OTHER-WINDOW is non-nil (use <b>C-u</b>), display in another window.</li> </ul>
Editing REXX Code	REXX support is minimal. Aside font locking and keyword abbreviations, return handling, indentation are available. <ul style="list-style-type: none"> <li>In my fork, as used by PEL, the code also supports the menu and Speedbar support to list REXX files along with their list of procedure definitions. I also added the commands to move point to the beginning of the next or previous REXX procedure.</li> </ul>		
Indent REXX code	<tab>	(rexx-indent-command &optional WHOLE-EXP)	Indent the current line as REXX code.
Move to the beginning of current do or select statement	C-c C-d	(rexx-find-matching-do)	Set mark, look for the "do" or "select" for the present block.
Move point to next procedure definition	<ul style="list-style-type: none"> <li>C-c C-n</li> <li>&lt;f12&gt; &lt;down&gt;</li> </ul> <div>&lt;f11&gt; SPC R &lt;down&gt;</div>	(rexx-goto-next-procedure &optional N SILENT DONT-PUSH-MARK)	Move point to the Nth procedure forward. <ul style="list-style-type: none"> <li>Search is controlled by the value of ‘rexx-regexp-for-procedure’ user option. Skip over procedure definitions inside comments or string.</li> <li>If no valid procedure definition is found, don't move point, issue an error describing the failure unless SILENT is non-nil, in which case the function returns nil on error and non-nil on success. The error message states the number of instanced searched, the regexp used and the number of instances found.</li> <li>On success, the function push original position on the mark ring unless DONT-PUSH-MARK is non-nil.</li> <li>Shift marking is supported by:               <ul style="list-style-type: none"> <li><b>C-c C-n</b> when Emacs runs in graphics mode.</li> <li><b>&lt;f12&gt; &lt;down&gt;</b> in graphics and terminal mode.</li> </ul> </li> <li>Move to previous marked location with <b>C-u C-SPC, M-`</b> or <b>&lt;f6&gt;&lt;f6&gt;</b>.</li> </ul>
Move point to previous procedure definition	<ul style="list-style-type: none"> <li>C-c C-p</li> <li>&lt;f12&gt; &lt;up&gt;</li> </ul> <div>&lt;f11&gt; SPC R &lt;up&gt;</div>	(rexx-goto-previous-procedure &optional N SILENT DONT-PUSH-MARK)	Move point to the Nth procedure backward. <ul style="list-style-type: none"> <li>Search is controlled by the value of ‘rexx-regexp-for-procedure’ user option. Skip over procedure definitions inside comments or string.</li> <li>If no valid procedure definition is found, don't move point, issue an error describing the failure unless SILENT is non-nil, in which case the function returns nil on error and non-nil on success. The error message states the number of instanced searched, the regexp used and the number of instances found.</li> <li>On success, the function push original position on the mark ring unless DONT-PUSH-MARK is non-nil.</li> <li>Shift marking is supported by:               <ul style="list-style-type: none"> <li><b>C-c C-p</b> when Emacs runs in graphics mode.</li> <li><b>&lt;f12&gt; &lt;up&gt;</b> in graphics and terminal mode.</li> </ul> </li> <li>Move to previous marked location with <b>C-u C-SPC, M-`</b> or <b>&lt;f6&gt;&lt;f6&gt;</b>.</li> </ul>
Debug a REXX file	C-C C-c	(rexx-debug PATH ARGS)	Run a rexx program FILE in buffer *rexx-FILE*.           The directory containing FILE becomes the initial working directory and source-file directory.  This code is not complete and not tested. 

### REXX— References

Document	Notes
REXX Programming Language	
The REXX Programming Language - Wikipedia	
The REXX Language Association	
rexx.org - Mark Hessling's page	This site has links to several REXX tools.
REXX is Still the King - Mark Damon Hughes blog	Provides some interesting info on REXX. Written in 2019.
REXX Implementations	
Regina REXX Interpreter homepage	Homepage of an open source REXX interpreter that has been ported to several OS that is 100% ANSI REXX compliant.
Regina REXX Interpreter source Mirror @ GitHub	
Homebrew regina-rexx formula for macOS	Install it on macOS with: brew install regina-rexx
Homebrew regina-rexx formula for Linux	
Emacs support for REXX	
rexx-mode @ emacsattic @ GitHub	
prouleau/rexx-mode @ GitHub	The version used by PEL: it supports customization, imenu/speedbar list of REXX files and their procedures, commands to move to the next/previous procedure(s).
Open Object REXX	Object Oriented REXX
Open Object REXX @ Wikipedia	
Open Object REXX Documentation	Links to several manuals both in HTML and PDF format
Open Object REXX Downloads @ Sourceforge	

Document	Notes
<a href="#">NetRexx</a>	A REXX language derivative that integrates ideas from Object Rexx and Java. Free License, open source. Runs on the JVM. See <a href="#">¶¶ - NetRexx</a>
NetRexx @ Wikipedia	
<a href="#">Emacs support for NetRexx</a>	
<a href="#">netrexx-mode.el @ netrexx.org</a>	A 2003 implementation that supports Net-Rexx on Emacs...
<a href="#">netrexx-mode.el @ GitHub</a>	...and its mirror in GitHub.
<a href="#">prouleau/netrexx-mode.el @ GitHub</a>	... and my updated version, on GutHub, that fixes byte-compiler warnings, used by PEL.