






## Emacs support for REXX

Description	Keystroke	Function	Note
<b>REXX Programming Language Support</b>	Support for the <b>REXX programming language</b> is minimal: you can activate the rexx-mode package by setting the <b>pel-use-rexx</b> 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 <b>rexx-mode</b> external package  automatically downloaded and installed by PEL when the <b>pel-use-rexx</b> user option is set to <b>t</b>.               <ul style="list-style-type: none"> <li>PEL adds the following regular expression to <b>auto-mode-alist</b> 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 <b>netrexx-mode.el</b> external package.</li> </ul>		
<b>Open this PDF file.</b> 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>	<b>(pel-help-pdf &amp;optional OPEN-WEB-PAGE)</b>	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 <b>pel-flip-help-pdf-arg</b> 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>	<b>(pel-customize-pel &amp;optional OTHER-WINDOW)</b>	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>	<b>(pel-customize-library &amp;optional OTHER-WINDOW)</b>	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>
<b>Editing REXX Code</b>	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>		
<b>Indent REXX code</b>	<div>&lt;tab&gt;</div>	<b>(rexx-indent-command &amp;optional WHOLE-EXP)</b>	Indent the current line as REXX code.
<b>Move to the beginning of current do or select statement</b>	<div>C-c C-d</div>	<b>(rexx-find-matching-do)</b>	Set mark, look for the "do" or "select" for the present block.
<b>Move point to next procedure definition</b>	<div> <ul style="list-style-type: none"> <li>C-c C-n</li> <li>&lt;f12&gt; &lt;down&gt;</li> </ul> </div> <div>&lt;f11&gt; SPC R &lt;down&gt;</div>	<b>(rexx-goto-next-procedure &amp;optional N SILENT DONT-PUSH-MARK)</b>	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>
<b>Move point to previous procedure definition</b>	<div> <ul style="list-style-type: none"> <li>C-c C-p</li> <li>&lt;f12&gt; &lt;up&gt;</li> </ul> </div> <div>&lt;f11&gt; SPC R &lt;up&gt;</div>	<b>(rexx-goto-previous-procedure &amp;optional N SILENT DONT-PUSH-MARK)</b>	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>
<b>Debug a REXX file</b>	<div>C-C C-c</div>	<b>(rexx-debug PATH ARGS)</b>	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
<b>REXX Programming Language</b>	
<b>The REXX Programming Language - Wikipedia</b>	
<b>The REXX Language Association</b>	
<b>rexx.org - Mark Hessling's page</b>	This site has links to several REXX tools.
<b>REXX is Still the King - Mark Damon Hughes blog</b>	Provides some interesting info on REXX. Written in 2019.
<b>REXX Implementations</b>	
<b>Regina REXX Interpreter homepage</b>	Homepage of an open source REXX interpreter that has been ported to several OS that is 100% ANSI REXX compliant.
<b>Regina REXX Interpreter source Mirror @ GitHub</b>	
<b>Homebrew regina-rexx formula for macOS</b>	Install it on macOS with: <code>brew install regina-rexx</code>
<b>Homebrew regina-rexx formula for Linux</b>	
<b>Emacs support for REXX</b>	
<b>rexx-mode @ emacsattic @ GitHub</b>	
<b>prouleau/rexx-mode @ GitHub</b>	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).
<b>Open Object REXX</b>	Object Oriented REXX
<b>Open Object REXX @ Wikipedia</b>	
<b>Open Object REXX Documentation</b>	Links to several manuals both in HTML and PDF format
<b>Open Object REXX Downloads @ Sourceforge</b>	

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.