↖ Topic Index

graphviz-dot—mode: Graphviz Dot Mode 🚧

Description	Keystroke	Function	Note

Graphviz Dot	📦 With the **graphviz-dot-mode package** external package installed, Emacs can support **Graphviz-Dot** markup files and can also render the resulting graphic files. PEL downloads, installs and activates it when the **pel-use-graphviz-dot** user option is set to **t**.
	• By default, Emacs treats files with the .dot and .gv extensions as **DOT** files.
	  • With PEL, you can add more file types by adding the association to the **pel-auto-mode-alist** user option.
	  • PEL adds these to Emacs **auto-mode-alist** user-option on startup.		

Open this PDF file. See also: ∑ **Help/Info**	**<f11> SPC M-g <f1>** **<f12> <f1>**	(**pel-help-pdf** &optional OPEN-WEB-PAGE)	Open the M **Graphviz Dot** local PDF. If the prefix argument (like **C-u** or **M--**) is used, then it opens the remote GitHub hosted raw PDF instead. If the **pel-flip-help-pdf-arg** user-option is set it's the other way around.
Customize PEL Graphviz Dot Support See also: ∑ **Customize**	**<f11> SPC M-g <f2>** **<f12> <f2>**	(**pel-customize-pel** &optional OTHER-WINDOW)	Customize PEL Graphviz-Dot support. • If OTHER-WINDOW is non-nil (use **C-u**), display in another window.
Customize Emacs Graphviz Dot Support See also: ∑ **Customize**	**<f11> SPC M-g <f3>** **<f12> <f3>**	(**pel-customize-library** &optional OTHER-WINDOW)	Customize Emacs Graphviz-Dot support. • If OTHER-WINDOW is non-nil (use **C-u**), display in another window.
Activate Graphviz Dot mode	**<f11> <f5> M-g**	(**graphviz-dot-mode**)	Toggle use of the graphviz-dot-mode to edit Graphviz Dot files or comments using graphviz-dot syntax.

Editing Dot syntax	The following commands help writing Graphviz DOT code. See the following reference documents:
	• Documentation, which contains the following and much more:
	  • The DOT Language
	  • Node, Edge and Graph Attributes
	  • Node Shapes
	  • Arrow Shapes
	  • Colors		

Indent current line See also: ∑ **Indentation**	**<tab>**	(**indent-for-tab-command** &optional ARG) (**graphviz-dot-indent-line**)	Indent current line of dot code (regardless of point position within the line)
Indent the graph statement at point	• **C-M-q** • **<f12> <tab>**	(**graphviz-dot-indent-graph**)	Indent the graph/digraph/subgraph where point is at. • This will first reach the beginning of the graph were point is at, and then indent this and each subgraph in it.
Break line and indent	**M-j**	(**indent-new-comment-line** &optional SOFT)	Break line at point and indent, continuing comment if within one. • This indents the body of the continued comment under the previous comment line. • This command is intended for styles where you write a comment per line, starting a new comment (and terminating it if necessary) on each line. • If a fill column is specified, it overrides the use of the comment column or comment indentation. • The inserted newline is marked hard if variable 'use-hard-newlines' is true, unless optional argument SOFT is non-nil.
Compiling			
Compile the graph	• **C-c C-c** • **<f12> c**	(compile COMMAND &optional COMINT)	Compile the graph using the program identified by graphviz-dot-dot-program to the output format identified by graphviz-dot-preview-extension. The command is shown in the mini buffer, allowing modification before execution.
Jump to next match	• **C-x `** • **M-g n** • **M-g M-n**	(**next-error** &optional ARG RESET)	A prefix ARG specifies how many error messages to move; negative means move back to previous error messages. Just C-u as a prefix means reparse the error message buffer and start at the first error.
Jump to previous match	• **M-g p** • **M-g M-p**	(**previous-error** &optional N)	Prefix arg N says how many error messages to move backwards (or forwards, if negative).
Viewing			
Preview image in Emacs buffer	• **C-C C-p** • **<f12> p**	(**graphviz-dot-preview**)	Compile the graph and preview it in an other buffer. ☛ ⚠ Only works in graphics mode.
View image using external program	**C-c C-v**	(**graphviz-dot-view**)	Run an external viewer. • This creates an external process every time it is executed. If 'graphviz-dot-save-before-view' is set, the current buffer is saved before the command is executed.
Preview Graphics	If you have code comments in Graphviz-Dot format inside a file of any major mode, you can use the following command to render the graphics.		

Preview diagram created from Graphviz DOT markup embedded in comments
 	
See also:
• M **Graphviz Dot**
	<f11> <f5> G	(**pel-render-commented-graphviz-dot** &optional POS)	Render the Graphviz-Dot markup embedded in current mode comment.
			Search at POS if specified, otherwise search around point.
			Use region if identified otherwise use Graphviz-Dot block.
			☝ The graphviz DOT code must be located within a block delimited by the following special keywords (that are also in comments):
			  • **@start-gdot**
			  • **@end-gdot**
			⚠ The current implementation leaves the created image file in a temporary directory. You will probably want to move that file or delete it, otherwise the size of this directory will increase with each of these created files. The file names use the pel-gdot- prefix.
			📦 Requires the **graphviz-dot-mode package** external package, activated by **pel-use-graphviz-dot** user option set to **t**.

Graphviz Dot Variables			
graphviz-dot-dot-program	Program used to compile the graphs. Default := 'dot'.		
graphviz-dot-preview-extension	File type to use for output. Default := 'png'.		
graphviz-dot-view-command	Command to run when 'graphviz-dot-view' is executed. Default:= 'dotty %s'		
graphviz-dot-view-edit-command	If the user should be asked to edit the view command. Default:= nil		
graphviz-dot-save-before-view	Automatically save current buffer berore 'graphviz-dot-view'.		
graphviz-dot-mode-hook	Hook variable for initialization of the mode.		

Graphviz Dot Mode — References

Description & URL	Notes
Graphviz Dot	
GraphViz @ Wikipedia	Overview description with various links.

GraphViz.org - Graphviz Graph Visualization Software	Graphviz home page. Includes:
	• Gallery
	• Documentation, which contains the following and much more:
	  • The DOT Language
	  • Command-line Usage
	  • Output formats
	  • Node, Edge and Graph Attributes
	  • Node Shapes
	  • Arrow Shapes
	  • Colors
	• Resources

Graphviz Dot Mode for Emacs	
graphviz-dot-mode @ GitHub	