Emacs support for the Seed7 Programming Language

Description	Keystroke	S support for the Seed / P		Note			
Seed7 Editing		d7 programming language is experimental and not yet	documented except fo				
O Help & customization	The seed7-mode ext	ernal package is installed when 🛂 the pel-use-seed7		· · · · · · · · · · · · · · · · · · ·			
CommentsTemplate Expansion	Seed7 files are files with .sd7 and .s7i extensions. The seed7-mode supports:						
o Seed7 abbreviations	Seed7 code highlighting	ng					
 <u>Auto-indent</u> <u>Marking</u>		k or line-end comments. Ability to select which type is command to select the comment style allowing easy s		swim. les of multi-line comments, a feature provided by Emacs that			
O Navigation	PEL uses and provide	des an easy selection at prompt.	· ·				
CompilationSeed7 Information		across function and procedures as well to start/end of g use of all imenu-based navigation commands and pro-		s/procedure as well as enum and struct. s callable (functions and procedures), interfaces, enums, structs.			
	1	and top menu with available commands. (see <u>Menus</u>	1)				
Last updated on:	 Seed7-syntax-aware auto-indentation and auto-fill-mode are supported. Code keyword expansion to Seed7 statements with ability to jump to next field to fill with tempo markers and navigation to those. 						
	 outline-minor-mode to list the name of Seed7 callables. See <u>Dutline</u> for more information. Invocation of Seed7 compiler tools to perform static analysis or compilation of Seed7 code. 						
	2025-06-09						
Open this PDF file.	<f11> SPC 7 <f1></f1></f11>	(pel-help-pdf &optional OPEN-WEB-PAGE)	Open the All - Seed7	local PDF. If the prefix argument (like C-u or M) is used,			
See also: <u>∑ Help/Info</u>	<f12> <f1></f1></f12>			ote GitHub hosted raw PDF instead. If the pel-flip-help-pdf-			
T Customine DEI	45115 and 7 4505	(sel customine sel gentional OTLIFD WINDOW)	arg user-option is set it's the other way around.				
∑ Customize PEL Seed7 support	<f11> SPC 7 <f2></f2></f11>	(pel-customize-pel &optional OTHER-WINDOW)	Customize PEL Seed7 support. • If OTHER-WINDOW is non-nil (use C-u), display in another window.				
	<f12> <f2></f2></f12>						
∑ Customize Emacs Seed7 support	<f11> SPC 7 <f3></f3></f11>	(pel-customize-library &optional OTHER-WINDOW) Customize Emacs Seed7 support: seed7 • If OTHER-WINDOW is non-nil (use C-u), display in another window		• •			
Seed? Support	<f12> <f3></f3></f12>		TOTHER-WINDOW IS HOT-IIII (use C-u), display in another window.				
Comments		navailable to insert and manipulate commands, listed in	∑ Comments. Some	are duplicated here for convenience.			
	The ones specific to seed						
Toggle between Seed7	C-c ;	(seed7-toggle-comment-style &optional ARG)		nment style between block and line comments. .RG, if supplied, switches to block comment style when positive,			
(* block *) and			to line comment style when negative, and just toggles it when zero or left out.				
# line style				e for all Seed7 buffers is controlled by the `seed7-uses-block- ble user-option. The default is line style comments.			
Insert, realign,	M-;	M-; (comment-dwim ARG)		Insert or realign comment on current line (or region if a region is active).			
comment/uncomment	"		On a single line, the comment is placed after the code.				
region			• C-u M-; execute				
Toggle display of comments in buffer or	<f11> ; ;</f11>	(hide/show-comments-toggle &optional START END)	Toggle hiding/showing of comments in the active region or whole buffer. • If the region is active then toggle in the region. Otherwise, in the whole buffer.				
active region		· · · · · · · · · · · · · · · · · · ·		This requires the <u>hide-comnt.el</u> package (see <u>S Comments</u>).			
See also: <u>E Comments</u>			when the pel-use-hide-comnt user option is t .				
Change comment	<f11> ; s</f11>	(pel-comment-style &optional CUSTOMIZE)	Select a comment style for the buffer: prompts with the list of available styles, showing the currently used one. Apply the choice to the current buffer. • With C-u prefix, open the customize buffer to control selection of the default				
style for buffer							
	comment style for all buffers (the comment-style user option).						
	Emacs supports several comment styles, as specified by the comment-styles user-option (which can be modified). Some of these styles only take effect when a						
	region of several lines is comments. By changing the style you can create the boxed comments, for instance and also uncomment the box comment with comment-swim (bound to M-;) and then change for another comment style in the same buffer.						
	 The style selected by the command only affects the current buffer. It is not persistent. The persistent setting is the comment-style user option. 0 = plain: Start in column 0 (do not indent), as in Emacs-20 						
As of Emacs 30, Emacs	• 1 = indent-or-triple: Start in column 0, but only for single-char starters						
supports 8 different comment styles, listed	• 2 = indent: Full comment per line, ends not aligned • 3 = aligned: Full comment per line, ends aligned						
here: ➡➡	 4 = box: Full comment per line, ends aligned, + top and bottom 5 = extra-line: One comment for all lines, end on a line by itself 						
	• 6 = multi-line: One comment for all lines, end on last commented line						
_	• 7 = box-multi: One comment for all lines, + top and bottom						
Code Template	The <u>seed7-mode</u> supports a set of code keyword expansion to Seed7 statements with ability to jump to next field to fill with <u>tempo markers</u> and navigation to these fields to complete the template easily.						
expansion	Code keyword expansion is performed by the seed7-complete-statement-or-indent command, bout to the <tab> key.</tab>						
	 Type the keyword then type <tab> to expand the keyword into the corresponding code that will be properly indented.</tab> There are 2 groups of supported keywords. 						
	• The keywords shown in the first part of the table expand to their corresponding code template when the keyword is the only word on the line and point is placed just after the last keyword character.						
Top level or block	inc	include statement	for	for statement			
declarations.	const	constant declaration	foru	for-until statement			
Type the keyword at the	var proc	variable declaration	fors	for-step statement for-each statement			
beginning of the line	func	procedure declaration function declaration	fore foreu	for-each statement combined with an until condition			
and hit <tab> to expand the</tab>	funcs	short function declaration	forek	for-each-key statement			
corresponding code.	enum struct	enum type declaration struct type declaration	foreku fork	for-each-key statement combined with an until condition for-key statement			
	case	case statement	forku	for-key statement combined with an until condition			
	if	if statement	repeat	repeat - until statement			
	ife ifei	if statement with an else clause if statement with an elsif clause	while	while statement			
	ifeie	if statement with an elsif and an else clause					
Dorom star			oing neverthered:	pro use to expend the measurements of a least time.			
Parameter declarations	in The second group of key	words are expanded when the keyword precedes a clo Declaration of an in-parameter.	callbn	Declaration of a call-by-name parameter.			
Also expand with <tab></tab>	inout	Declaration of an <u>in-parameter</u> . Declaration of an <u>inout-parameter</u> .	ref	Declaration of a <u>can-by-name parameter</u> . Declaration of a <u>reference-parameter</u> .			
	invar	Declaration of an in-var-parameter.	val	Declaration of a value-parameter.			
Expand keyword or indent	<tab></tab>	(seed7-complete-statement-or-indent)		I code keyword properly located, this perform code expansion,			
			 leaving point at the first location that must be filled. In that case you can then type <backtab> to move to the next field that needs to be filled (or has already been filled). Those are tempo markers that stay in the buffer until the buffer is closed.</backtab> 				
				where else indent the line or selected block.			
Move to next field	<backtab></backtab>	(tempo-forward-mark)	Move point to the nex	t tempo marker, the next template field to fill.			

Description	<u>Keystroke</u>	Fur	nction		<u>Note</u>	
Seed7-specific abbreviations	The <u>seed7-mode</u> supports Seed7-specific abbreviations for <u>Emacs abbrev-mode</u> when the <u>seed7-support-abbrev-mode</u> customizable user-option is on (the default). • All abbreviation and their text expansion are set by the <u>seed7-abbreviations</u> customizable user-option list.					
See also: <u> \[\tilde{\Sigma} \] Abbreviations</u>	 All abbreviation and their text expansion are set by the seed/-abbreviations customizable user-option list. The default list is shown below. All abbreviations start with a semi-colon. You can modify the default and add other abbreviations through customization. These abbreviations are system abbreviations, treated specially by the abbrev-mode in the sense that youcanngt modify them dynamically via the abbrev-mode commands. But you don't need to since they can be modified by customization. Of course you can create other abbreviations that help you write code or comments. See <u>Sabbreviations</u> for more details related to abbreviations in Emacs. To expand the abbreviations, the <u>abbrev-mode</u> must be active: type the abbreviation followed by a word-separating character, such as <space>, <ret>, semi-colon, period, comma, etc</ret></space> 					
Pragmas & in-	р	ragmas	in-statement keywords		in-middle statement keywords	
statement keywords	;de	decls	;fo	forward	;dt	downto
	;in	info	;n	new	;exc	exception
	;li	library	;no	noop	;lo	local
	;msg	message	;ra	raise	;pa	param
	;na	names	;rt	return	;rg	range
	;syn	syntax			;rs	result
	;sys	system			;st	step
	;tr	trace			100	
Block clause			block	ause keywords		
keywords	;ct	catch	;e	else	;o	otherwise
		Calcii	;ei		;w	
D 16 11				elsif	,	when
Pre-defined types	;a	array	;db	database	;rat	rational
	;bi	bigInteger	;du	database	;rf	rational
	;br	bigRational	;en	duration	;rfl	reference
	;b3	bin32		enum		ref_list
	;b6	bin64	;fi	expr	;s	set
	;bt			file	;sq	sqlStatement
		bitset	;fs	fileSys	;sti	string
	;bo	boolean	;fl	float	;stu	struct
	;bs	bstring	;h	hash	;tx	text
	;ca	category	;i	integer	;ti	time
	;c	char	;ob	object	;ty	type
	;cf	clib_file	;pro	process	;v	void
	;co	color	;pr	program	;pw	PRIMITIVE_WINDOW
	;cx	complex				
Pre-defined constants				ined constants		
	;em	empty	;f	FALSE	;inf	Infinity
			;t	TRUE		
Pre-defined variables			pre-de	fined variables		
	;ck	CONSOLE_KEYBOARD	;sc	STD_CONSOLE	;sn	STD_NULL
	;gk	GRAPH_KEYBOARD	;se	STD_ERR	;so	STD_OUT
	;kb	KEYBOARD	;si	STD_IN		
Errinfo values			erri	info values		
	;ok	OKAY_NO_ERROR	;dse	DESTROY_ERROR	;me	MEMORY_ERROR
	;ae	ACTION_ERROR	;fe	FILE_ERROR	;ne	NUMERIC_ERROR
	;ce	COPY_ERROR	;ge	GRAPHIC_ERROR	;oe	OVERFLOW_ERROR
	;cre	CREATE_ERROR	;ie	INDEX_ERROR	;re	RANGE_ERROR
	;dbe	DATABASE_ERROR	;ine	IN_ERROR		
Syntax-aware outomatic ndentation	Unless explicitly disabled by setting the seed7-auto-indent user-option to nil, the <tab></tab> and <return></return> key perform syntax-aware automatic indentation of Seed7 code. The <return></return> key also supports the auto-fill-mode. • The number of columns used for each indentation level is controlled by the seed7-indent-width user-option, which defaults to 2. • Emacs can use hard tabs as appropriate when you activate the indent-tabs-mode . If it is off Emacs only uses space characters. See the <u>S Indentation</u> page for more information related to indentation control and commands.					
Auto-fill-mode	The seed7-mode supports Emacs auto-fill-mode, useful when typing comments. See the Fill/Justify page and the pel-comment-style command above.					
Marking	The sood? made supr	port specialized marking. It is	also compatible with other	Emancs native and nad	ckage commands. See <u>E Marki</u>	ng for more information

Mark current callable

C-M-h

(seed7-mark-defun)

Mark the current Seed7 function or procedure.

Put the mark at the end and point at the beginning.

If point is before or between 2 functions or procedure, mark the next one.

Description	<u>Keystroke</u>	Function	<u>Note</u>		
Code Navigation	The seed7-mode supports syntax-aware procedure/function as well as block aware navigation commands PEL provides some extra key bindings to Emacs native navigation commands. The seed7-mode also supports imenu-compliant parsing which enables the ability to use a large set of navigation packages. See navigation by symbol definition in the Navigation page for more information. The seed7-mode navigation commands display the name and type of block found when the seed7-verbose-navigation user-option is turned on (set to t).				
Shift-Selection	If you press and hold the shift key while typing a movement command, that sets the mark before moving point (Emacs name for cursor) so that the region extends from the original point to its new position. This is called: Shift-Selection . • Shift selection is supported by some navigation commands, not all. The following symbols are used to identify whether the command supports shifts selection: • ↓ This command supports shift selection in GUI and terminal mode. • ↓ This command supports shift selection in GUI mode. • ↓ This command supports shift selection in GUI mode and also in terminal mode under some conditions (described in the description cell for the command). • ‡ This command does not support shift selection. Sometimes for this you can first set the mark before moving. • Pressing the Shift key when using the key binding for commands that do not show any of these 3 arrows have no impact on the shift selection (and may be inappropriate for the command).				
Move Point	The following sub-sections describe how to navigate across various types of textual and syntactical entities.				
• by <u>defun</u>	The commands move point by Seed7 function and procedure definitions. In PEL: The <f12> cursor key mappings use <up> and <down> to move to the beginning or end of the function, procedure or other blocks. The <f6> cursor key mapping use <up> and <down> to move to the beginning or end of the function or procedure. The <f6> cursor key mapping use <right> and <left> to move to the beginning or end of the next/previous function or procedure. The advantage of the <f6> and <f12> key bindings is they support Shift-Selection for Emacs in terminal mode, as opposed to the key bindings that sue the Control key which can only support Shift-Selection when Emacs is running in Graphics mode.</f12></f6></left></right></f6></down></up></f6></down></up></f12>				
Backward to beginning of defun	• <f6> <up> • C-M-a • C-M-<home> • C-[C-a • Esc C-a</home></up></f6>	(seed7-beg-of-defun &optional N SILENT DONT-PUSH-MARK)	Move backward to the beginning of a defun. With ARG, do it that many times. Negative ARG means move forward to the ARGth following beginning of defun. Prints the name of the function or procedure in the message area. On successful move, you can move back to original position by typing M-`, <f6> or <f11> . Supports Shift-Selection in graphics mode. <f6><up> supports it in terminal mode too.</up></f6></f11></f6>		
Forward to end of defun ♥	• <f6> <down> • C-M-e • C-M-<end> • C-[C-e • Esc C-e</end></down></f6>	(seed7-end-of-defun &optional N SILENT DONT-PUSH-MARK)	Move forward to next end of defun. With argument, do it that many times. Negative argument -N means move back to Nth preceding end of defun. Prints the name of the function or procedure in the message area. On successful move, you can move back to original position by typing M-`, <f6> <f6> or <f11> . Supports Shift-Selection in graphics mode. <f6><down> supports it in terminal mode too.</down></f6></f11></f6></f6>		
Forward to start of next defun	<f6> <right></right></f6>	(seed7-beg-of-next-defun &optional N SILENT DONT-PUSH-MARK)	Move forward to the beginning of the next function or procedure. • With optional argument N, repeat the search that many times. • Move back to previous position with M-`, <f6> <f6> or <f11> . • Supports Shift-Selection.</f11></f6></f6>		
Backward to end of previous define with will be replaced	<f6> <left></left></f6>	(pel-end-of-previous-defun &optional SILENT DONT-PUSH_MARK)	Move backwards to the end of the previous function definition. Issue user error not find end of previous function unless SILENT is non-nil. If the end of previous function is found, push the start location to the mark ring unless DONT-PUSH_MARK is non-nil. Move back to previous position with M-`, <f6> <f6> or <f11> . Supports Shift-Selection.</f11></f6></f6>		
Forward to end of current block statement	<f12> <down></down></f12>	(seed7-to-block-forward)	Move forward from the beginning of a Seed7 block to its end. Supports the Seed7 if/end if, block/end block, case/end case, enum/end enum, for/end for, repeat/until, struct/end struct, while/end while. It also supports moving to the end of a function or a procedure. Move back to previous position with M-`, <f6> <f6> or <f11>. Supports Shift-Selection.</f11></f6></f6>		
Backward to beginning of current block statement	<f12> <up></up></f12>	(seed7-to-block-backward)	Move backward from the end of a Seed7 block to its beginning. supports the Seed7: if/end if, block/end block, case/end case, enum/end enum, for/end for, repeat/until, struct/end struct, while/end while. It also supports moving to the end of a function or a procedure. Move back to previous position with M-`, <f6> <f6> or <f11> . supports Shift-Selection.</f11></f6></f6>		
Compilation	The Seed7 source code is either interpreted or compiled. In both cases you can verify it's validity by performing a static check of the code, an operation that does not generate any binary file but perform the same language checking that the compiler will do.				
Static check or compile Seed7 file	<f12> c</f12>	(seed7-compile &optional COMPILE)	Static check current Seed7 file, show errors in compilation-mode buffer. • If optional COMPILE argument set, compile the file to executable instead.		
See <u>See Compilation</u> Mode	 For example: type C-u <f12> c for compiling the file. Without the C-u prefix it just static checks the file, an operation that is much faster.</f12> The static analysis is performed by the command identified by the seed7-checker user-option, which defaults to s7-check. You can specify any command with or without its path. The compilation is performed by the command identified by the seed7-compiler user-option, which defaults to s7c. You can specify any command with or without its path. Any detected error is shown in a *compilation* <u>S Compilation Mode</u> buffer. Use it to navigate to the line of the code in error. 				

Emacs & Seed7 — References

Document	Notes		
The Seed7 Programming Language		ed7 Manual ed7 Language Reference	
	Seed7 @ reddit Seed7 @ Rosetta code		
Presentations	 The Seed7 Programming Language @ Youtube The Seed7 Programming Language Presentation at CPP Vienna @ Youtube Another speech about the Seed7 Programming Language 		
	Modern Extensible Languages. Daniel Zingaro, McMaster U. April 11, 2007 (pdf)		
Emacs support 🚧 is partial, not yet completed.	seed7-mode @ Github		
Other tools that support Seed7	 ripgrep a very fast grep replacement - supports seed7 file types with this pull request accepted April 7 2025 With this version of ripgrep, you can use deadgrep to identify Seed7 files by name in Emacs. See <u>Carep</u> ugrep another very fast grep replacement - supports seed7 files with this pull request. 		