PEL Topics Index

Note: with PEL, type `<f11> <f1>` to open this PDF index.

Emacs Reference Cards

☝With PEL you can access these via the `<f11> ? e r` key sequence.
See Σ Help/Info

These are links to the PDF version of official English version of the quick reference cards for GNU Emacs and popular external packages. PEL documents Emacs key bindings as well, these cards provide useful complement to what PEL provides.

Emacs	Calc	Gnus	Magit Cheatsheet	Org	Viper
Emacs survival card	Dired	Gnus booklet	Magit Ref-card		VIP

➢ PEL Overview

- PEL repo
- PEL Readme
- PEL Manual
- PEL NEWS

This table holds links to the PEL file tables. Each cell holds a hyperlink to the GitHub hosted raw PDF table.

☝For the best user experience, use a browser that can render PDF directly instead of downloading.
- Mozilla Firefox (version > 78) does that perfectly. You may need to activate a plug-in for other browsers.
- With that in place, you can browse through all the PDFs and reach a vast amount of information quickly.

☝From within Emacs open this topic index PDF by typing the `<f11> ? <f1>` key sequence. More help topics with `<f11> ? p` keys.

☝The symbols, **colour** **coding** and various other conventions are described in the ➢Legend PDF.

• General Information.	➢Legend	➢Recommended Emacs User Option	➢Themes		
• Development Information	➢PEL	iMenu/Speedbar support	PEL Naming Conventions		
• Migration Guide	➢CRiSP ⇔ Emacs				

OS Desktop Key Bindings

(Bindings that don't clash with PEL)

macOS Fct Keys	macOS Keys	Ubuntu 16.04 Desktop Keys		
	terminal settings	Mint 20 Desktop Keys		

Feature Comparisons

Completion Modes Compatibility | Speedbar/iMenu Mode Compatibility | Shells/Terminals Comparisons

Key Prefixes & Suffixes

Σ Modifier Keys | ΣNumkeypad | ➢PEL | Keys - Fn | Keys - F11

Σ Emacs Features

- A Guided Tour of Emacs.
- Awesome-Emacs
- MELPA and GNU ELPA

The PEL tables named at right describe Emacs commands & key bindings for concepts & features. The cell color is light-blue for major-mode, Light-red for minor mode.
Emacs commands can be executed by name or bound to key sequences. The commands may have ***arguments*** and keys can express them.
- Emacs Keys
- Numeric Arguments

You can also:
- Run Command by Name

Emacs uses a concept of modes:
- Emacs Major and Minor Modes
  - Major Modes
  - Minor Modes
  - Choosing Modes

PEL provides key sequences to toggle minor modes.

Cells link titles starting with only Σ are Emacs generic features, blue links are external packages. The green links are mostly PEL extensions.

Σ Abbreviations	Σ Diff & Merge	Σ Grep	Σ Marking	Σ Scrolling	T Templates
Σ Align	Σ Dired	Σ Help/Info	Σ Menus	Σ Search/Replace	Σ Text Modes
Σ Auto-Completion	Σ Display - Lines	Σ Hide/Show	Σ Mode Line	Σ Sessions	Σ Time Tracking
Σ Autosave/Backup	Σ Drawing	Σ Highlight (colors)	Σ Mouse	Σ start Shells/REPLs	Σ Transpose
Σ Bookmarks	Σ Enriched Text	Σ ibuffer-mode	Σ Narrowing	Σ shell-mode	ΣŦ Treemacs
Σ Buffers	Σ Faces/Fonts	Σ Indentation	Σ Navigation	Σ term-mode	Σ Undo/Redo
Σ Case Conversions	ΣP Fast Startup	Σ Input Method	Σ Outline	Σ vterm-mode	Σ VCS-Git ŦMagit
Σ Close/Suspend	Σ File-mngt	Σ Inserting Text	Σ Packages	ΣŦ Smartparens	Σ VCS-Mercurial
Σ Comments	Σ File/Dir Variables	Σ Key-Chords	ΣŦ Projectile	Σ Sorting	Σ VCS-Subversion
Σ Completion/Input	Σ Fill/Justify	Σ Keyboard Macros	Σ Rectangles	Σ Speedbar	Σ Web
Σ Counting	Σ Frames	ꝑŦ- Lispy	Σ Registers	Σ Spell Checking	Σ Whitespace
ΣM CUA				Σ SyntaxCheck	Σ Windows
Σ Cursor					Σ Xref - Cross Refs
Σ Customize					
Σ Cut & Paste					

ŦꝑŁ - Emacs Lisp concepts & tools: Ł ERT (Emacs Lisp Regression Testing) | Ł Hooks | Ł※ - Emacs Lisp Types

XRef - Cross Reference Tools

See also: Σ Xref

Emacs supports various cross reference mechanisms described in the Σ Xref table. These mechanisms take advantage of various external tools and integrate with them. Notes about those tools are available in the tables listed in this section.

Xref-Support | Xref-Backend

Build Tools & Preprocessor

PEL supports installation and partial setup of the following tools:

PEL has support for several build tools but they are not all documented in a page.
- Nix — Requires nix-mode external package — activated when pel-use-nix-mode user-option is tuned on.
- Tup — Requires tup-mode external package — activated when pel-use-tup user-option is tuned on.

ꝑŁ - M4	ꝑŁ - Make gmake				

Command Line Scripting Languages: bash, sh, zsh

Data Serialization

Ⓓ CWL | Ⓓ YAML

Utility: GNU readline

Data Modelling/ Specification

Ⓢ ASN.1 asn1-mode | Ⓢ MIB snmp-mode | Ⓢ YANG

Hardware Description Languages

Verilog future | VHDL future

Text Markup Languages

Ⓜ AsciiDoc | Ⓜ Markdown | Ⓜ Org-Mode | Ⓜ reStructuredText

OS App Control Scripting Languages

• Graphics Markup

Ⓜ Graphviz Dot | Ⓜ MscGen | Ⓜ PlantUML

ꝑŁ- AppleScript

Programming Languages

Main Paradigm of Programming Language Families
- ***Actor Model:*** Ⓐ
- ***Concatenative*** Ⓚ
- ***Concurrent:*** ⓒ
- ***Functional:*** ⓕ ***Pure:*** Ⓕ
- ***Imperative:*** ⓘ ***or no token***
- ***Object Oriented*** ∞
- ***Has Syntactic Macros:*** ⓜ

- The programming languages supported by PEL are listed here in alphabetical order.
- Emacs (and PEL) also provides basic support for other programming languages not listed here.

Future support for Crystal, Elm, Kotlin, Lua, Purescript, ReasonML, Seed7, Typescript, Zig and documentation of support for Ada, Fortran, Javascript, Java, Modula, Pascal (based on my need for them or requests (if any)).

Emacs has major mode support for several programming languages. PEL currently adds extra support for some of them, listed below.

BEAM Programming Languages	*Functional Languages*	Javascript target	Lisp Family Languages	Lisp-like Languages	
Curly Bracket Languages	Java Virtual Machine Languages	ML Family Languages	Scheme Language Dialects	Stack Based Languages	

The following lists the programming languages in alphabetical order.
- The cell colours give a coarse indication of the programming language family(ies).

Ada future	ꝑŁ - D ⓘⓕⒶ	ꝑŁ - Gambit ⓕⓜ	ꝑŁ - Janet ⓘⓕⓜ	Objective-C future	Scala future
ꝑŁ - Arc ⓕⓜ	Dart future	ꝑŁ - Gerbil ⓕⓜⒶ	Java future	ꝑŁ - OCaml ⓘⓕ	ꝑŁ - Scheme ⓕⓜ
ꝑŁ - C	Eiffel future	ꝑŁ - GNU Guile ⓕⓜ	ꝑŁ - Javascript	Pascal future	Seed7 future
ꝑŁ - C++	ꝑŁ - Elm future Ⓕ	ꝑŁ - Gleam	ꝑŁ - Julia ⓜ	ꝑŁ - Perl	Swift future
ꝑŁ - Chez ⓕⓜ	ꝑŁ - Elixir ⓒⓜⓕⒶ	ꝑŁ - Go	Kotlin future	ꝑŁ - Python	ꝑŁ - Tcl future ⓕⓘ
ꝑŁ - Chibi ⓕⓜ	ŁꝑŁ - Emacs Lisp	Groovy future	ꝑŁ - LFE ⓒⓜⓕⒶ	ꝑŁ - Purescript Ⓕ	ꝑŁ - Typescript
ꝑŁ - Chicken ⓕⓜ	ꝑŁ - Erlang ⓒⓕⒶ	ꝑŁ - Haskell Ⓕ	Lua future	ꝑŁ - Racket ⓕⓜ	ꝑŁ - UNIX Shell
ꝑŁ - Clojure ⓕⓜ	Factor Ⓚⓕ∞ⓜ	Haxe future	Modula future	ꝑŁ - ReasonML	ꝑŁ - V
Common Lisp ⓕⓜ	ꝑŁ - Forth Ⓚ	ꝑŁ - Hy *(python)* ⓜ	ꝑŁ - NetRexx	ꝑŁ - REXX	Zig future
Crystal future	Fortran future		ꝑŁ - Nim ⓜ	ꝑŁ - Ruby	
				ꝑŁ - Rust	