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 ⅀ 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
- Discussions

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

⅀ ⌨Modifier Keys	⅀ ⌨Numkeypad	➢PEL	⌨Keys - Fn	⌨Keys - F11

⅀ Emacs Features

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

☝ Run Emacs daemon & client on macOS

The PEL tables listed 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 ⅀ are Emacs generic features, blue links are external packages. The green links are mostly PEL extensions.

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

𝔈𝔓𝔩 - Emacs Lisp concepts & tools	𝔈 display-buffer	𝔈※ - ELisp Types	𝔈 ERT (regr-testing)	𝔈 Hooks		

XRef - Cross Reference Tools

See also: ⅀ Xref

Emacs supports various cross reference mechanisms described in the ⅀ 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-Frontend	🚦 Xref-Backend			

PEL supports installation and partial setup of the following tools:

Build Tools & Preprocessor

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.

𝔓𝔩 - M4	𝔓𝔩 - Make gmake			

Command Line Scripting Languages:

bash, sh, zsh

Utility: GNU readline

Data Serialization	Ⓓ CWL	Ⓓ YAML	
Data Modelling/ Specification	Ⓢ ASN.1 asn1-mode	Ⓢ MIB snmp-mode	Ⓢ YANG
Hardware Description Languages	Verilog 🚧future	VHDL 🚧future	

Text Markup Languages	Ɱ AsciiDoc	Ɱ Markdown	Ɱ Org-Mode	Ɱ reStructuredText
Graphics Markup	Ɱ Graphviz Dot	Ɱ MscGen	Ɱ PlantUML	

OS App Control Scripting Languages

𝔓𝔩 - AppleScript

Programming Languages

Main Paradigm of Programming Language Families
- ***Actor Model:*** Ⓐ
- ***Concatenative*** Ⓚ
- ***Concurrent:*** Ⓒ
- ***Functional:*** ⓕ ***Pure:*** Ⓕ
- ***Imperative:*** ⓘ ***or no token***
- ***Object Oriented*** ꝏ
- ***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	𝔓𝔩 - D ⓘⓕⒶ	𝔓𝔩 - Gambit ⓕⓜ	𝔓𝔩 - Janet ⓘⓕⓜ	Objective-C 🚧future	Scala 🚧future
𝔓𝔩 - Arc ⓕⓜ	Dart 🚧future	𝔓𝔩 - Gerbil ⓕⓜⒶ	Java 🚧future	𝔓𝔩 - OCaml ⓘⓕ	𝔓𝔩 - Scheme ⓕⓜ
𝔓𝔩 - C	Eiffel 🚧future	𝔓𝔩 - GNU Guile ⓕⓜ	𝔓𝔩 - Javascript 🚧	Pascal 🚧future	Seed7 🚧future
𝔓𝔩 - C++	𝔓𝔩 - Elm 🚧future Ⓕ	𝔓𝔩 - Gleam	𝔓𝔩 - Julia ⓜ	𝔓𝔩 - Perl	Swift 🚧future
𝔓𝔩 - Chez ⓕⓜ	𝔓𝔩 - Elixir ⒸⓜⓕⒶ	𝔓𝔩 - Go	Kotlin 🚧future	𝔓𝔩 - Python	𝔓𝔩 - Tcl 🚧future ⓕⓘ
𝔓𝔩 - Chibi ⓕⓜ	𝔈𝔓𝔩 - Emacs Lisp	Groovy 🚧future	𝔓𝔩 - LFE ⒸⓜⓕⒶ	𝔓𝔩 - Purescript Ⓕ	𝔓𝔩 - Typescript 🚧
𝔓𝔩 - Chicken ⓕⓜ	𝔓𝔩 - Erlang ⒸⓕⒶ	𝔓𝔩 - Haskell Ⓕ	Lua 🚧future	𝔓𝔩 - Racket ⓕⓜ	𝔓𝔩 - UNIX Shell
𝔓𝔩 - Clojure ⓕⓜ	Factor Ⓚⓕꝏⓜ	Haxe 🚧future	Modula 🚧future	𝔓𝔩 - ReasonML 🚧	𝔓𝔩 - V
Common Lisp ⓕⓜ	𝔓𝔩 - Forth Ⓚ	𝔓𝔩 - Hy *(python)* ⓜ	𝔓𝔩 - NetRexx	𝔓𝔩 - REXX	Zig 🚧future
Crystal 🚧future	Fortran 🚧future		𝔓𝔩 - Nim ⓜ	𝔓𝔩 - Ruby	
				𝔓𝔩 - Rust	