PEL Topics 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 quickly 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	Σ🖮 Numkeypad	➢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 the Emacs commands and key bindings for generic Emacs concepts and features.
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.

The links that start with only Σ Emacs generic features, the blue links are external packages. The green links are mostly PEL extensions.

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

𝔵℘ɪ - Emacs Lisp concepts & tools	𝔵 ERT (Emacs Lisp Regression Testing)	𝔵 Hooks	𝔵※ - 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. 🚧 This is work in progress.

Xref-Support	Xref-Backend

Build Tools & Preprocessor

PEL supports installation and partial setup of the following tools:
- 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.

℘ɪ - M4	℘ɪ - Make

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

Ⓜ AsciiDoc	Ⓜ Markdown	Ⓜ Org-Mode	Ⓜ reStructuredText

• Graphics Markup

Ⓜ Graphviz Dot	Ⓜ MscGen	Ⓜ PlantUML

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.
- PEL also provides basic support for other programming languages not listed here.
- Emacs supports other programming languages directly, 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	Command Line Scripting Languages
Curly Bracket Languages	Java Virtual Machine Languages	ML Family Languages	Scheme Language Dialects	Stack Based Languages	OS App Control Scripting Languages

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

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