↖ Topic Index

Emacs support for Rust 🚧

Description	Keystroke	Function	Note
Rust Programming Language Support PEL Rust support activation ➽ Σ **Indentation** control ➽ Last updated on:			PEL activates **Rust** support when the **pel-use-rust** user-option is turned on. PEL supports the **rust-mode** and the **rust-ts-mode**. PEL supports **Rust** via **rust-mode** when **pel-use-rust** is **t** and via **rust-ts-mode** when **pel-use-rust** user options is set to with-tree-sitter. • PEL only support Tree-Sitter mode on **Emacs >= 30**, when **pel-use-tree-sitter** is set to t. See Σ **Tree Sitter** • Speedbar support for Rust files listing functions and types. See Σ **Speedbar** for more info about it. PEL provide support for the Rust programming language and its various implementations by providing access to the following external packages: 📦 The **rust-mode** external package. PEL activates it when the **pel-use-rust-mode** user-option is turned on (**t**). 📦 The **rustic** external package. PEL activates it when the **pel-use-rustic** user-option is turned on (**t**). 📦 The **flycheck-rust** external package. PEL activates it when the **pel-use-flycheck-rust** user-option is is turned on (**t**). 📦 The **emacs-racer** external package. PEL activates it when the **pel-use-emacs-racer** user-option is turned on (**t**). 📦 The **cargo** external package. PEL activates it when the **pel-use-cargo** user-option is turned on (**t**). 🎛 Rust indentation is controlled by the following user-options: • **rust-indent-offset** sets the number of columns used for indentation. It defaults to 4. • PEL sets **tab-width** with the same value in rust buffers so that manual indentation commands use the same number of columns to indent. • **pel-rust-use-tabs** controls whether hard tabs are used for indentation (nil by default). • PEL sets **indent-tabs-mode** with the value of **pel-rust-use-tabs** in rust buffers. 2025-10-15
Open this PDF file. See also: Σ **Help/Info**	`<f11> SPC r <f1>` `<f12> <f1>`	(**pel-help-pdf** &optional OPEN-WEB-PAGE)	Open the **PEL - Rust** 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.
Σ **Customize** PEL Rust support	`<f11> SPC r <f2>` `<f12> <f2>`	(**pel-customize-pel** &optional OTHER-WINDOW)	Customize PEL Rust support. • If OTHER-WINDOW is non-nil (use **C-u**), display in another window.
Σ **Customize** Emacs Rust support	`<f11> SPC r <f3>` `<f12> <f3>`	(**pel-customize-library** &optional OTHER-WINDOW)	Customize Emacs Rust support: rust-mode, rustic, racer, cargo. • If OTHER-WINDOW is non-nil (use **C-u**), display in another window.
Show PEL setup for Rust	`<f12> ?` `<f11> SPC r ?`	(**pel-rust-setup-info** &optional APPEND)	Display Rust setup information inside a *pel-rust-info* buffer with buttons providing quick access to the customization buffer of each variable shown. The information shown includes the value and interpretation of: • pel-use-rust (whether the classic or tree-sitter based major mode is used). • the user options controlling format on save, indentation and hard tab width rendering. To append information in the buffer instead of clearing the previous content type any prefix argument (such as **C-u**) before the command keystroke.
Cargo run	`<f12> c`	(**rust-run**)	Build the Rust file using Cargo and run it.
Add/Remove the dbg! macro	`<f12> d`	(**rust-dbg-wrap-or-unwrap**)	Either remove or add the dbg! macro.
Run Clippy, Rust Lint Checker	`<f12> l`	(**rust-run-clippy**)	Run 'cargo clippy'.

Emacs & Rust — References

Document	Notes
Fancy Rust development with Emacs	May 2016. Describes how to use rust-mode
rust-mode: A major Emacs mode for editing Rust source code	A GitHub site
rust-mode	See: http://julienblanchard.com/2016/fancy-rust-development-with-emacs/
Racer for emacs	
company-mode ; Modular in-buffer completion framework for Emacs	
Why Rust?	Safari book online
rust-cross	This GitHub site states: Everything you need to know about compiling rust programs!
Taking Rust everywhere with rustup	A Rust site blog on rustup
Cross compiling Rust on OS X for Raspberry Pi 3	March 2016 article on cross compiling Rust on Raspberry Pi3
Raspberry Pi Bare Metal Programming with Rust	
Rust source code	