
















# Emacs support for Rust ⚠️

Description	Keystroke	Function	Note
<b>Rust Programming Language Support</b>	 This is an early version of this page. More information will be provided soon.		
	 PEL activates <b>Rust</b> support when the <b>pel-use-rust</b> user-option is turned on (t). PEL provide support for the Rust programming language and its various implementations by providing access to the following external packages:		
	 The <b>rust-mode</b> external package.	 PEL activates it when the <b>pel-use-rust-mode</b> user-option is turned on (t).	
	 The <b>rustic</b> external package.	 PEL activates it when the <b>pel-use-rustic</b> user-option is turned on (t).	
	 The <b>flycheck-rust</b> external package.	 PEL activates it when the <b>pel-use-flycheck-rust</b> user-option is turned on (t).	
	 The <b>emacs-racer</b> external package.	 PEL activates it when the <b>pel-use-emacs-racer</b> user-option is turned on (t).	
	 The <b>cargo</b> external package.	 PEL activates it when the <b>pel-use-cargo</b> user-option is turned on (t).	
<b>Open this PDF file.</b> See also:  <b>Help/Info</b>	<b>&lt;f11&gt; SPC r &lt;f1&gt;</b>	<b>(pel-help-pdf &amp; optional OPEN-WEB-PAGE)</b>	Open the local copy of this <b>pel - Rust</b> PDF file unless a command prefix (like <b>C-u</b> ) was used. In that case it opens the Github-hosted file instead.
	<b>&lt;f12&gt; &lt;f1&gt;</b>		
 <b>Customize</b> PEL Rust support	<b>&lt;f11&gt; SPC r &lt;f2&gt;</b>	<b>(pel-customize-pel &amp; optional OTHER-WINDOW)</b>	Customize PEL Rust support. • If OTHER-WINDOW is non-nil (use <b>C-u</b> ), display in another window.
	<b>&lt;f12&gt; &lt;f2&gt;</b>		
 <b>Customize</b> Emacs Rust support	<b>&lt;f11&gt; SPC r &lt;f3&gt;</b>	<b>(pel-customize-library &amp; optional OTHER-WINDOW)</b>	Customize Emacs Rust support: rust-mode, rustic, racer, cargo. • If OTHER-WINDOW is non-nil (use <b>C-u</b> ), display in another window.
	<b>&lt;f12&gt; &lt;f3&gt;</b>		
<b>Cargo run</b>	<b>&lt;f12&gt; c</b>	<b>(rust-run)</b>	Build the Rust file using Cargo and run it.
<b>Add/Remove the dbg! macro</b>	<b>&lt;f12&gt; d</b>	<b>(rust-dbg-wrap-or-unwrap)</b>	Either remove or add the dbg! macro.
<b>Run Clippy, Rust Lint Checker</b>	<b>&lt;f12&gt; l</b>	<b>(rust-run-clippy)</b>	Run ‘cargo clippy’.

## Emacs & Rust — References

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