



























































































































































🚦 Tree-Sitter parsers for Emacs 🚧🚧🚧

TreeSitter parsers	Supported by PEL	User-option control	tree-sitter mode	Language grammar	With  iMenu support	With  Speedbar support	Status					
<div>Last updated on: 2025-10-15</div> <div>See Also:  <a href="#">Tree Sitter</a></div>	Indicates yes only when explicitly supported by PEL code.	The name and value of PEL user option that control whether Tree-Sitter aware mode is used.	The name of the major mode command that supports the tree-sitter based control. Modes names in black are built-in Emacs.	Name and link to the project providing the language grammar.	Whether all commands based on imenu work in tree-sitter mode.	Whether Speedbar support works for the tree-sitter based mode.	Identify any known problem here. Later this will be expanded to several features	 As PEL introduces explicit support for more major mode, new class will be filled. Once enough tree-sitter support is explicitly implemented, I will add explicit support for LSP and then check the support of various features like completion, navigation based on LSP and tree-sitter. I will then add more columns related to these features here and in the  <a href="#">Language Servers</a> table.				
 - <a href="#">Ada</a>  												
 - <a href="#">AppleScript</a>												
<a href="#">APL</a> 												
 - <a href="#">Arc</a> 												
 - <a href="#">awk</a> 												
 - <a href="#">C</a> 												
 - <a href="#">C++</a> 												
Carbon  future 												
 - <a href="#">Chez</a> 												
 - <a href="#">Chibi</a> 												
 - <a href="#">Chicken</a> 												
 - <a href="#">Clojure</a> 												
<a href="#">Common Lisp</a> 												
<a href="#">Crystal</a> 												
 - <a href="#">D</a>  												
<a href="#">Dart</a> 												
 - <a href="#">Eiffel</a>  												
 - <a href="#">Elm</a>  												
 - <a href="#">Elixir</a>  	Yes	pel-use-elixir	elixir-ts-mode	<a href="#">tree-sitter-langs</a> ➡ <a href="#">tree-sitter-elixir</a>	Yes	Yes	OK					
 - <a href="#">Emacs Lisp</a>												
 - <a href="#">Erlang</a>  												
 - <a href="#">Factor</a>  												
 - <a href="#">Forth</a> 												
<a href="#">Fortran</a> 												
 - <a href="#">Gambit</a> 												
 - <a href="#">Gerbil</a>  												
 - <a href="#">GNU Guile</a> 												
 - <a href="#">Gleam</a>	Yes	See note ➡	<a href="#">gleam-ts-mode</a>	<a href="#">tree-sitter-langs</a> ➡ <a href="#">tree-sitter-gleam</a>	Yes	Yes	OK	Note: Gleam is only supported by a Tree-Sitter aware mode. There's no classic mode for Gleam.				

TreeSitter parsers	Supported by PEL	User-option control	tree-sitter mode	Language grammar	With  iMenu support	With  Speedbar support	Status					
 <b>Go</b>	Yes	pel-use-go	go-ts-mode	<a href="#">tree-sitter-langs</a> ➔ <a href="#">tree-sitter-go</a>	Yes	Yes	OK					
 <b>Go</b> go.mod	Yes	pel-use-go	go-mod-ts-mode	<a href="#">tree-sitter-go-mod</a>	Yes	Yes	OK					
Groovy 												
 <b>Haskell</b> 												
Haxe 												
 <b>Hy</b> (python) 												
 <b>Janet</b>   												
Java 												
 <b>JavaScript</b> 												
 <b>Julia</b> 												
Kotlin 												
 <b>LFE</b>    												
 <b>Lua</b>   												
 <b>Modula</b>												
 <b>NetRexx</b>												
 <b>Nim</b>  												
 <b>Objective-C</b> 												
 <b>OCaml</b>  												
 <b>Odin</b> 												
 <b>Pascal</b>												
 <b>Perl</b> (perl5)												
 <b>Pike</b>   												
 <b>Python</b>    												
 <b>Purescript</b>  												
      												
 <b>Racket</b>  												
 <b>ReasonML</b> 												
 <b>REXX</b>												
 <b>Ruby</b>												
 <b>Rust</b> 	Yes	pel-use-rust	rust-ts-mode	<a href="#">tree-sitter-langs</a> ➔ <a href="#">tree-sitter-rust</a>	Yes	Yes	OK					
Scala 												
 <b>Scheme</b>  												
 <b>Seed7</b>    												

TreeSitter parsers	Supported by PEL	User-option control	tree-sitter mode	Language grammar	With  iMenu support	With  Speedbar support	Status					
 - Smalltalk  @												
 - Swift												
 - Tcl  ①①												
 - Typescript 												
 - UNIX Shell												
 - V												
 - Zig 	Yes	pel-use-zig	<a href="#">zig-ts-mode</a>	<a href="#">tree-sitter-langs</a> ➡ <a href="#">tree-sitter-zig</a>	Yes	Yes	<ul style="list-style-type: none"> <li>• fortification does not work</li> <li>• incomplete indentation control</li> <li>• no format on save like zig-mode</li> </ul>					