



































































































































































🚦 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:  Tree Sitter</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  Language Servers table.				
 - Ada  												
  - AppleScript												
APL 												
 - Arc 												
 - awk 												
 - C 												
 - C++ 												
Carbon  future 												
 - Chez 												
 - Chibi 												
 - Chicken 												
 - Clojure 												
Common Lisp 												
Crystal 												
 - D  												
Dart 												
 - Eiffel   												
 - Elm  												
 - Elixir   	Yes	pel-use-elixir	elixir-ts-mode	tree-sitter-langs ➡ tree-sitter-elixir	Yes	Yes	OK					
  - Emacs Lisp												
 - Erlang  												
 - Factor   												
 - Forth 												
Fortran 												
 - Gambit 												
 - Gerbil  												
 - GNU Guile 												
 - Gleam	Yes	See note ➡	gleam-ts-mode	tree-sitter-langs ➡ tree-sitter-gleam	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					
 Go 	Yes	pel-use-go	go-ts-mode	tree-sitter-langs ➡ tree-sitter-go	Yes	Yes	OK					
 Go go.mod	Yes	pel-use-go	go-mod-ts-mode	tree-sitter-go-mod	Yes	Yes	OK					
Groovy 												
 Haskell 												
Haxe 												
 Hy (python) 												
 Janet   												
Java 												
 Javascript 												
 Julia 												
Kotlin 												
 LFE    												
 Lua   	Yes	pel-use-lua	lua-ts-mode	tree-sitter-langs ➡ tree-sitter-lua 	Yes	Yes	<ul style="list-style-type: none"> fortification does not work The tree-sitter-lua project used by tree-sitter-langs seems unmaintained. It should probably use tree-sitter-grammars/tree-sitter-lua 					
 Modula												
 NetRexx												
 Nim  												
 Objective-C 												
 OCaml  												
 Odin 												
 Pascal												
 Perl (perl5)												
 Pike   												
 Python    												
 Purescript  												
     												
 Racket  												
 ReasonML 												
 REXX												
 Ruby												
 Rust 	Yes	pel-use-rust	rust-ts-mode	tree-sitter-langs ➡ tree-sitter-rust	Yes	Yes	OK					
Scala 												

TreeSitter parsers	Supported by PEL	User-option control	tree-sitter mode	Language grammar	With  iMenu support	With  Speedbar support	Status					
 - Scheme  												
 -Seed7    												
 -Smalltalk  												
 -Swift												
 - Tcl  												
 - Typescript 												
 - UNIX Shell												
 - V												
 -Zig 	Yes	pel-use-zig	zig-ts-mode	tree-sitter-langs ➡ tree-sitter-zig	Yes	Yes	<ul style="list-style-type: none"> • fortification does not work • incomplete indentation control • no format on save like zig-mode 					