















































🚦 Tree-Sitter parsers for Emacs 🚧🚧🚧

<u>TreeSitter parsers</u>	Supported by PEL	User-option control	tree-sitter mode	Language grammar	With <u>ΣiMenu</u> support	With <u>Σ Speedbar</u> support	Status				
Last updated on: 2025-10-15 See Also: <u>Σ Tree Sitter</u>	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 <u>🚦 Language Servers</u> table.			
📄 - Ada 🚧🚧🚧 ➡️📄											
📄🍏 - AppleScript											
APL 🚧🚧🚧											
📄 - Arc (f)📄											
📄 - awk (d)											
📄 - C 📄											
📄 - C++ @📄											
Carbon 🚧🚧🚧 future 📄											
📄 - Chez (f)📄											
📄 - Chibi (f)📄											
📄 - Chicken (f)📄											
📄 - Clojure (f)📄											
Common Lisp (f)📄											
Crystal 🚧🚧🚧											
📄 - D (i)(f)A											
Dart 🚧🚧🚧											
📄 - Eiffel 🚧🚧🚧 @ 📄											
📄 - Elm 🚧🚧🚧 (F)											
📄 - Elixir (c)(m)(f)A	Yes	pel-use-elixir	elixir-ts-mode	<a href="#">tree-sitter-langs ➡️ tree-sitter-elixir</a>	Yes	Yes	OK				
🔪📄 - Emacs Lisp											
📄 - Erlang (c)(f)A											
📄 - Factor (K)(f) @📄											
📄 - Forth (K)											
Fortran 🚧🚧🚧											
📄 - Gambit (f)📄											
📄 - Gerbil (f)📄A											
📄 - GNU Guile (f)📄											
📄 - Gleam	Yes	See note ➡️	<a href="#">gleam-ts-mode</a>	<a href="#">tree-sitter-langs ➡️ 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>   	Yes	pel-use-lua	lua-ts-mode	<a href="#">tree-sitter-langs</a> ➡ <a href="#">tree-sitter-lua</a> 	Yes	Yes	<ul style="list-style-type: none"><li>fortification does not work</li><li>The tree-sitter-lua project used by tree-sitter-langs seems unmaintained. It should probably use <a href="#">tree-sitter-grammars/tree-sitter-lua</a></li></ul>				
 <b>Modula</b>											
 <b>NetRexx</b>											
 <b>Nim</b>  	No: As of Emacs 30.2 there is no nim-ts-mode implemented yet.			<a href="#">alviss/tree-sitter-nim</a>	No	Yes, but since iMenu is not supported, nothing shows.	Does not exists yet.				
 <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>	Yes	pel-use-ruby	ruby-ts-mode	<a href="#">tree-sitter-langs</a> ➡ <a href="#">tree-sitter/tree-sitter-ruby</a>	Yes	Yes	OK				

TreeSitter parsers	Supported by PEL	User-option control	tree-sitter mode	Language grammar	With  iMenu support	With  Speedbar support	Status				
 - Rust	Yes	pel-use-rust	rust-ts-mode	<a href="#">tree-sitter-langs</a> ➡ <a href="#">tree-sitter-rust</a>	Yes	Yes	OK				
Scala 											
 - Scheme											
 - Seed7	No: As of Emacs 30.2 there is no seed7-ts-mode implemented yet.			No grammar exists yet.	Yes, for seed7-mode	Yes, for seed7-mode	Does not exists yet.				
 - Smalltalk											
 - Swift											
 - Tcl	No: As of Emacs 30.2 there is tcl-ts-mode implemented yet, even though the Tree-Siter grammar exists.			<a href="#">tree-sitter-langs</a> ➡ <a href="#">tree-sitter/tree-sitter-tcl</a>	Yes, for tcl-mode	Yes, for tcl-mode	Does not exists yet.				
 - 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>				