Abstract

Parser combinators are domain-specific languages that allow users to write parsers in a high-level manner strongly resembling their original grammar. However, as with any large framework, there are idiomatic ways to write clean and maintainable code using these libraries. New users may not be aware of these design patterns or struggle to apply them effectively in the correct contexts.

This project addresses this issue for the `parsley` parser combinator library in Scala, by developing an accompanying linter named `parsley-garnish`. Unlike the majority of linters, which are designed to detect generic issues in general-purpose languages, `parsley-garnish` is tailored to the specific idioms of parser combinators.

`parsley-garnish` is the first domain-specific linter for parser combinators. This project presents a set of lint rules designed to improve the quality of `parsley` codebases, and to help users write idiomatic parsers using established design patterns. Notably, `parsley-garnish` is able to automatically refactor left-recursive parsers into a correct and idiomatic form that `parsley` would otherwise be unable to handle.

Finally, this project also demonstrates and implements the infrastructure required to write complex domain-specific lint rules that transform parser representations in a high-level and declarative manner. This allows future lint rules to be written with ease, and for `parsley-garnish` to be extended with more powerful automatic code fixes in the future.