

miniclj

User Manual

Contents

1	About the language	1
1.1	Differences and limitations compared to Clojure	2
2	Data types	2
2.1	Numbers	2
2.2	Strings	2
2.3	Lists	2
2.4	Vectors	2
2.5	Maps	2
2.6	Sets	2
2.7	Nil	2
3	Callables	2
3.1	Collection functions	2
3.1.1	Access	2
3.1.2	Creation	3
3.1.3	Generation	3
3.1.4	Modification	3
3.1.5	Transducers	3
3.2	Comparison operations	3
3.3	Conditionals	3
3.4	Cycles	3
3.5	Factor operations	3
3.6	Grouping functions	3
3.7	I/O functions	3
3.8	Lambda functions	3
3.9	Scope functions	3
3.10	Typecasting functions	3

1 About the language

This project's aim is to create a compiler and virtual machine for a lisp-based language with similar semantics to Clojure. The base functions and data structures will be supported, and they must be accessible either through a Command-Line Interface or inside a web context.

1.1 Differences and limitations compared to Clojure

Other than not including a broader standard library compared to Clojure, miniclj has some differences and limitations, like:

- Support for symbols during runtime isn't supported because they must be linked to a memory address during compilation
- Expressions and lists are evaluated eagerly, miniclj doesn't support lazy sequences
- Lambda functions don't capture their enclosing environment/scope
- Support for macros wasn't implemented
- Code is strictly single threaded, and there is no support for using concurrency controls like atoms or promises

2 Data types

2.1 Numbers

2.2 Strings

2.3 Lists

2.4 Vectors

2.5 Maps

2.6 Sets

2.7 Nil

3 Callables

3.1 Collection functions

3.1.1 Access

`first`

```
(first collection)
```

Returns the first item in the collection. If the collect

- 3.1.2 Creation**
- 3.1.3 Generation**
- 3.1.4 Modification**
- 3.1.5 Transducers**
- 3.2 Comparison operations**
- 3.3 Conditionals**
- 3.4 Cycles**
- 3.5 Factor operations**
- 3.6 Grouping functions**
- 3.7 I/O functions**
- 3.8 Lambda functions**
- 3.9 Scope functions**
- 3.10 Typecasting functions**