# Clojure

- A History of Clojure
- clojure lib
- clojure guides
- Threading Macros Guide
- clojure web framework
- Learning Clojure

```
- brew install openjdk
- brew install leiningen
```

## Create Project

- clojure-cli
- leiningen

## Library

- fulcro
- quil
- http : hato

## Syntax

- Atomic Data Types
- Data Structures
- cheatsheet
- values
- clojure-by-example
- typed clojure
- 클로저 스타일 가이드

## Clojure is a Lisp

- Dynamic
- Code as data
- Reader
- Small core
- Sequences
- Syntactic abstraction

## 매크로

- Hygienic
- ->>

## 격언

- 클로저 프로그래머는 클로저로 애플리케이션을 작성하는 것이 아닐, 클로저로 애플리케이션을 작성하는 데 사용되는 언어를 작성하는 것이다.

## Learn

- https://practical.li/
- https://clojurebridge.org/
- Clojure from the ground up
- https://4clojure.oxal.org/
- Clojure Koans
- https://clojure.org/api/cheatsheet
- https://shaunlebron.github.io/t3tr0s-slides/#0
- https://github.com/unclebob/spacewar
- https://github.com/clojure-kr/clojure-complete
- https://johngrib.github.io/wiki/clojure/

## 클로저 참조 타입

| | ref | agent | atom | var |
|---|---|---|---|---|
| 조정 여부 | ☐ | | | |
| 비동기 | | ☐ | | |
| 반복 가능 여부 | ☐ | | ☐ | |
| 스레드-로컬 | | | | ☐ |

ref의 독특한 특징은 관리된다는 점이다. 다시 말하면 여러 참조들을 읽고 쓸 때 경합 조건이 발생하지 않도록 보장해준다는 뜻이다 비동기는 업데이트 요청은 나중에 다른 스레드에서 실행되도록 큐에 입력되고, 요청을 생성했던 스레드는 계속해서 동작하는 것을 의미한다. 반복 가능 여부는 참조 값 업데이트를 마치는 동작을 추정하여 반복할 수 있는지와 관련한 특성이다 마지막으로 스레드-로컬은 변경 정보를 단일 스레드의 상태에 격리시켜 스레드 안전성을 확보하는 것을 말한다

## Docs

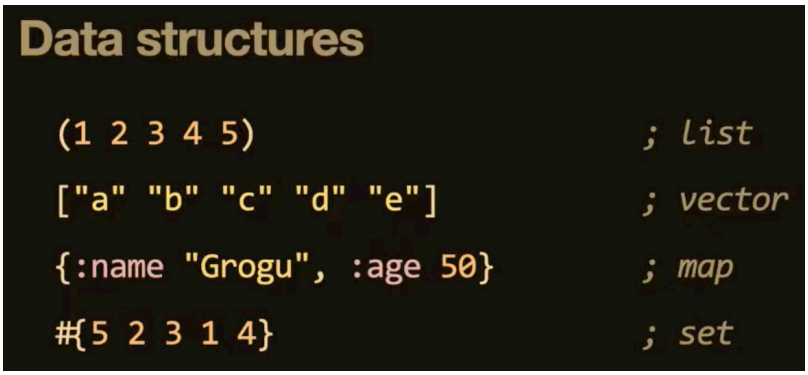
- Sequences
- Namespace
- File
- lib
- metadata
- destructuring
- shorthand

- macro
- gensym
- protocol
- dynamic var
- core collection functions
- Learn Clojure
- Clojure Web Framework
- Clojure Design Patterns
- unclebob_spacewar
- project.clj
- clojure.spec.alpha
- clojure syntax

# 책

- 클로저 시작하기
- The Joy Of Clojure
- Programming Clojure

### Data structures




### Code as data