In Clojure 1.3, How to read and write a file

65 ☆ 40

I'd like to know the "recommended" way of reading and writing a file in clojure 1.3 .

1. How to read the whole file
2. How to read a file line by line
3. How to write a new file
4. How to add a line to an existing file

Thanks.

file clojure io

share | improve this question

asked Oct 13 '11 at 15:54
jolly-san
507 ●1 ●5 ●7

1 First result from google: lethain.com/reading-file-in-clojure – jcubic Oct 13 '11 at 16:14

2 This result is from 2009, some things have been changed lately. – Sergey Oct 25 '11 at 21:00

Indeed. This StackOverflow question is now the first result on Google. – mydoghasworms Apr 18 at 7:11

add comment

4 Answers

active oldest votes

109 ✓

Assuming we're only doing text files here and not some crazy binary stuff.

Number 1: how to read an entire file into memory.

```
(slurp "/tmp/test.txt")
```

Not recommended when it is a really big file.

Number 2: how to read a file line by line.

```
(use 'clojure.java.io)
(with-open [rdr (reader "/tmp/test.txt")]
  (doseq [line (line-seq rdr)]
    (println line)))
```

The `with-open` macro takes care that the reader is closed at the end of the body. The reader function coerces a string (it can also do a URL, etc) into a `BufferedReader`. `line-seq` delivers a lazy seq. Demanding the next element of the lazy seq results into a line being read from the reader.

Number 3: how to write to a new file.

```
(use 'clojure.java.io)
(with-open [wrtr (writer "/tmp/test.txt")]
  (.write wrtr "Line to be written"))
```

or with `spit`, the opposite of `slurp`:

```
(spit "/tmp/test.txt" "Line to be written")
```

Again, with-open takes care that the `BufferedWriter` is closed at the end of the body. Writer coerces

tagged
file × 26601
clojure × 6477
io × 6169

asked 2 years ago
viewed 20346 times
active 2 months ago

Community Bulletin
event Annual user survey ends - make your voice heard! – in 13 hours

Linked
4 Writing to a file in Clojure
3 Read a file in clojure and ignore the first line?
2 Why is line-seq returning clojure.lang.Cons instead of clojure.lang.LazySeq?

Related

a string into a `BufferedWriter`, that you use use via java interop: `(.write wrtr "something").`

Number 4: append a line to an existing file.

```
(use 'clojure.java.io)
(with-open [wrtr (writer "/tmp/test.txt" :append true)]
  (.write wrtr "Line to be appended"))
```

Same as above, but now with append option.

Or again with `spit`, the opposite of `slurp`:

```
(spit "/tmp/test.txt" "Line to be written" :append true)
```

PS: To be more explicit about the fact that you are reading and writing to a File and not something else, you could first create a File object and then coerce it into a `BufferedReader` or Writer:

```
(reader (file "/tmp/test.txt"))
;; or
(writer (file "tmp/test.txt"))
```

The file function is also in clojure.java.io.

PS2: Sometimes it's handy to be able to see what the current directory (so ".") is. You can get the absolute path in two ways:

```
(System/getProperty "user.dir")
```

or

```
(-> (java.io.File. ".") .getAbsolutePath)
```

share | improve this answer edited Mar 22 at 11:33 answered Oct 13 '11 at 16:59

Michiel Borkent
10.3k ●8 ●38 ●65

Thank you very very much for your detailed answer. I'm glad to get to know the recommended way of File IO (text file) in 1.3. There seems to have been some libraries about File IO (clojure.contrb.io, clojure.contrib.duck-streams and some examples directly using Java BufferedReader FileInputStream InputStreamReader) which made me more confusing. Moreover there is little information about Clojure 1.3 especially in Japanese (my natural language) Thank you. – jolly-san Oct 14 '11 at 0:51

Hi jolly-san, tnx for accepting my answer! For your information, clojure.contrib.duck-streams is now deprecated. This possibly adds to the confusion. – Michiel Borkent Oct 14 '11 at 7:21

Very informative. Thanks. – octopusgrabbus Nov 14 '12 at 14:35

That is, `(with-open [rdr (reader "/tmp/test.txt")] (line-seq rdr))` yields `IOException Stream closed` instead of a collection of lines. What to do? Am getting good results with the answer by @satyagraha, though. – 0dB Oct 27 at 4:37

1 This has to do with lazyness. When you use the result of line-seq outside of the with-open, which happens when you print its result to the REPL, then the reader is already closed. A solution is to wrap the line-seq inside a doall, which forces evaluation immediately. `(with-open [rdr (reader "/tmp/test.txt")] (doall (line-seq rdr)))` – Michiel Borkent Oct 27 at 9:27

add comment

21

If the file fits into memory you can read and write it with slurp and spit:

```
(def s (slurp "filename.txt"))
```

(s now contains the content of a file as a string)

```
(spit "newfile.txt" s)
```

This creates newfile.txt if it doesnt exit and writes the file content. If you want to append to the file you can do

```
(spit "filename.txt" s :append true)
```

To read or write a file linewise you would use Java's reader and writer. They are wrapped in the namespace clojure.java.io:

316 How to create a Java String from the contents of a file?

10 Treating a file of Java floats as a lazy Clojure sequence

5 Make Clojure recognize and isolate the line in a file

6 A way to strip returned values from java.io.File.listFiles in Clojure

10 Read a very large text file into a list in clojure

19 Read, edit, and write a text file line-wise using Ruby

0 Lua file reading and writing error

3 Maintaining multiple open files for writing (Clojure)

4 How to read a file with test data in with Clojure?

1 LISP: How to read content from a file and write it in another file?

Hot Network Questions

I need a program where the user inputs an array of doubles and the program outputs the array sorted

Mathematical research published in the form of poems

Can We Decide Whether Small Computer Programs Halt?

more hot questions

```
(ns file.test
  (:require [clojure.java.io :as io]))

(let [wrtr (io/writer "test.txt")]
  (.write wrtr "hello, world!\n")
  (.close wrtr))

(let [wrtr (io/writer "test.txt" :append true)]
  (.write wrtr "hello again!")
  (.close wrtr))

(let [rdr (io/reader "test.txt")]
  (println (.readLine rdr))
  (println (.readLine rdr)))
; "hello, world!"
; "hello again!"
```

Note that the difference between slurp/spit and the reader/writer examples is that the file remains open (in the let statements) in the latter and the reading and writing is buffered, thus more efficient when repeatedly reading from / writing to a file.

Here is more information: slurp spit clojure.java.io Java's BufferedReader Java's Writer

share | improve this answer   edited Oct 13 '11 at 16:43   answered Oct 13 '11 at 16:08
Paul
2,830 ● 10 ● 19

1 Thank you Paul. I could learn more by your codes and your comments which are clear in the point focusing on answering my question. Thank you very much. – jolly-san Oct 14 '11 at 0:55

add comment

3

Regarding question 2, one sometimes wants the stream of lines returned as a first-class object. To get this as a lazy sequence, and still have the file closed automatically on EOF, I used this function:

```
(use 'clojure.java.io)

(defn read-lines [filename]
  (let [rdr (reader filename)]
    (defn read-next-line []
      (if-let [line (.readLine rdr)]
       (cons line (lazy-seq (read-next-line)))
       (.close rdr)))
    (lazy-seq (read-next-line)))
)

(defn echo-file []
  (doseq [line (read-lines "myfile.txt")]
    (println line)))
```

share | improve this answer   answered Dec 10 '12 at 13:52
satyagraha
118 ● 4

1 I don't think nesting `defn` is ideomatic Clojure. Your `read-next-line`, as far as I understand, is visible outside of your `read-lines` function. You might have used a `(let [read-next-line (fn [] ...))` instead. – kristianlm Oct 3 at 16:48

add comment

1

This is how to read the whole file.

If the file is in the resource directory, you can do this :

```
clojure (let [file-content-str (slurp (clojure.java.io/resource
"public/myfile.txt")])
```

remember to require/use clojure.java.io

share | improve this answer   edited Oct 3 at 16:59   answered Apr 22 at 13:23
DBD
4,898 ● 2 ● 18 ● 51
joshua
1,556 ● 2 ● 18 ● 28

add comment

Your Answer

Sign up or login

Sign up using Google

Sign up using Facebook

Sign up using Stack Exchange

Post as a guest

Name

Email

Post Your Answer

By posting your answer, you agree to the privacy policy and terms of service.

Not the answer you're looking for? Browse other questions tagged file clojure io or ask your own question.

question feed

about help badges blog chat data legal privacy policy jobs advertising info mobile **contact us** **feedback**

TECHNOLOGY			LIFE / ARTS	CULTURE / RECREATION	SCIENCE	OTHER
Stack Overflow	Programmers	Database Administrators	Photography	English Language & Usage	Mathematics	Stack Apps
Server Fault	Unix & Linux	Drupal Answers	Science Fiction & Fantasy	Skeptics	Cross Validated (stats)	Meta Stack Overflow
Super User	Ask Different (Apple)	SharePoint	Seasoned Advice (cooking)	Mi Yodeya (Judaism)	Theoretical Computer Science	Area 51
Web Applications	WordPress Answers	User Experience	Home Improvement	Travel	Physics	Stack Overflow Careers
Ask Ubuntu	Geographic Information Systems	Mathematica	**more (13)**	Christianity	MathOverflow	
Webmasters	Electrical Engineering	**more (14)**		Arqade (gaming)	**more (7)**	
Game Development	Android Enthusiasts			Bicycles		
TeX - LaTeX	Information Security			Role-playing Games		
				more (21)		