Cursor of HashMap records with RwLockGuard in Rust

Asked 3 years, 1 month ago
Active 3 years, 1 month ago
Viewed 662 times

6

2

I am new to Rust, and I am trying to implement a simple, thread-safe memory key-value store, using a `HashMap` protected within a `RwLock`. My code looks like this:

```
use std::sync::{ Arc, RwLock, RwLockReadGuard };
use std::collections::HashMap;
use std::collections::hash_map::Iter;

type SimpleCollection = HashMap<String, String>;

struct Store(Arc<RwLock<SimpleCollection>>);

impl Store {
  fn new() -> Store { return Store(Arc::new(RwLock::new(SimpleCollection::new()))) }

  fn get(&self, key: &str) -> Option<String> {
    let map = self.0.read().unwrap();
    return map.get(&key.to_string()).map(|s| s.clone());
  }

  fn set(&self, key: &str, value: &str) {
    let mut map = self.0.write().unwrap();
    map.insert(key.to_string(), value.to_string());
  }
}
```

So far, this code works OK. The problem is that I am trying to implement a `scan()` function, which returns a `Cursor` object that can be used to iterate over all the records. I want the `Cursor` object to hold a `RwLockGuard`, which is not released until the cursor itself is released (basically I don't want to allow modifications while a Cursor is alive).

I tried this:

```
use ...

type SimpleCollection = HashMap<String, String>;

struct Store(Arc<RwLock<SimpleCollection>>);

impl Store {
  ...

  fn scan(&self) -> Cursor {
    let guard = self.0.read().unwrap();
    let iter = guard.iter();
    return Cursor { guard, iter };
  }
}

struct Cursor<'l> {
  guard: RwLockReadGuard<'l, SimpleCollection>,
  iter: Iter<'l, String, String>
}

impl<'l> Cursor<'l> {
  fn next(&mut self) -> Option<(String, String)> {
    return self.iter.next().map(|r| (r.0.clone(), r.1.clone()));
  }
}
```

But that did not work, as I got this compilation error:

```
error[E0597]: `guard` does not live long enough
  --> src/main.rs:24:20
   |
24 |       let iter = guard.iter();
   |                  ^^^^^ borrowed value does not live long enough
25 |       return Cursor { guard, iter };
26 |   }
   |   - borrowed value only lives until here
   |
note: borrowed value must be valid for the anonymous lifetime #1 defined on the method body at 22:5...
  --> src/main.rs:22:5
   |
22 | /     fn scan(&self) -> Cursor {
23 | |       let guard = self.0.read().unwrap();
24 | |       let iter = guard.iter();
25 | |       return Cursor { guard, iter };
26 | |   }
   | |_____^
```

Any ideas?

rust rwlock

Share
Improve this question
Follow
edited Nov 23 '18 at 7:33

asked Nov 23 '18 at 1:40

Ayman Madkour
195 • 5

While `Cursor` is alive, you want nobody else to modify your `HashMap`? Is that the purpose of all that?
– hellow
Nov 23 '18 at 7:12

@hellow correct.
– Ayman Madkour
Nov 23 '18 at 7:22

1

Could you put your code into the playground so that we can more easily play around with it?
– Sebastian Redl
Nov 23 '18 at 7:55

1

@hellow I know. There's also this question and a Clippy lint. I personally don't use the `return` in the last statement, but don't feel strongly about what other people should do.
– Sven Marnach
Nov 23 '18 at 10:19

1

Possible duplicate of Why can't I store a value and a reference to that value in the same struct?
– Peter Hall
Nov 23 '18 at 14:21

Show 7 more comments

1 Answer

Active Oldest Votes

6

As mentioned in the comments, the problem is that structs generally can't be self-referential in Rust. The `Cursor` struct you are trying to construct contains both the `MutexGuard` and the iterator borrowing the `MutexGuard`, which is not possible (for good reasons – see the linked question).

The easiest fix in this case is to introduce a separate struct storing the `MutexGuard`, e.g.

```
struct StoreLock<'a> {
    guard: RwLockReadGuard<'a, SimpleCollection>,
}
```

On the `Store`, we can then introduce a method returning a `StoreLock`

```
fn lock(&self) -> StoreLock {
    StoreLock { guard: self.0.read().unwrap() }
}
```

and the `StoreLock` can expose the actual `scan()` method (and possibly others requiring a persistent lock):

```
impl<'a> StoreLock<'a> {
    fn scan(&self) -> Cursor {
        Cursor { iter: self.guard.iter() }
    }
}
```

The `Cursor` struct itself only contains the iterator:

```
struct Cursor<'a> {
    iter: Iter<'a, String, String>,
}
```

Client code first needs to obtain the lock, then get the cursor:

```
let lock = s.lock();
let cursor = lock.scan();
```

This ensures that the lock lives long enough to finish scanning.

Full code on the playground

Share
Improve this answer

Follow

answered Nov 23 '18 at 15:27

Sven Marnach

511k ● 113 ● 891 ● 798

Thanks a lot, @SvenMarnach. I guess I am still not used to the Rust mindset.
– Ayman Madkour
Nov 23 '18 at 15:51

Add a comment

Your Answer

Post Your Answer

By clicking "Post Your Answer", you agree to our terms of service, privacy policy and cookie policy

Not the answer you're looking for? Browse other questions tagged rust rwlock or ask your own question.

The Overflow Blog

- Sequencing your DNA with a USB dongle and open source code
- Don't push that button: Exploring the software that flies SpaceX rockets and...

Featured on Meta

- Providing a JavaScript API for userscripts
- Congratulations to the 59 sites that just left Beta

Linked

Why can't I store a value and a reference to that value in the same struct?

41
Why is using return as the last statement in a function considered bad style?

Related

336
How do I print the type of a variable in Rust?

Why doesn't println! work in Rust unit tests?

How to disable unused code warnings in Rust?

How do I split a string in Rust?

Rust package with both a library and a binary?

Factory method: instance does not live long enough

Problems with Tuple's lifetime in rust.

Why does the Rust compiler not optimize code assuming that two mutable references cannot alias?

Hot Network Questions

- What was the plutonium for, that was stolen at the start of The Amazing Spider-Man 2?
- Is there a deterministic guide to landing?
- How do I get the http endpoint to work for cardano-wallet?
- Changing a color of a link
- How to install a package via `apt-get` without flagging it as manually installed

more hot questions

Question feed

STACK OVERFLOW

Questions
Jobs
Developer Jobs Directory
Salary Calculator
Help
Mobile

PRODUCTS

Teams
Talent
Advertising
Enterprise

COMPANY

About
Press
Work Here
Legal
Privacy Policy
Terms of Service
Contact Us
Cookie Settings
Cookie Policy

STACK EXCHANGE NETWORK

Technology
Culture & recreation
Life & arts
Science
Professional
Business
API
Data

Blog
Facebook
Twitter
LinkedIn
Instagram