



Products

# borrowed value does not live long enough in loop

[Log in](#) [Sign up](#)

[Ask Question](#)

Asked 2 years, 1 month ago

Active 2 years, 1 month ago

Viewed 2k times



3



I'm trying to parse a file and return `Vec<Vec<&str>>` from a function. But I'm getting borrowed value error inside the file read loop while pushing to the vector.

```
use std::io::{self, BufReader, prelude::*};
use std::fs::File;

fn read() -> Vec<Vec<&static str>> {
    let file = File::open("~/test").expect("failed to read file");
    let reader = BufReader::new(file);
    let mut main_vector: Vec<Vec<&str>> = Vec::new();
    for line in reader.lines() {
        match line {
            Ok(v) => {
                let mut sub_vector: Vec<&str> = Vec::new();
                for element in v.split_whitespace().collect::<Vec<&str>>() {
                    sub_vector.push(element);
                }
                main_vector.push(sub_vector);
            },
            Err(e) => panic!("failed to parse: {:?}", e),
        }
    }
    //return main_vector;
}
```

Here's the compiler error:

```
error[E0597]: `v` does not live long enough
  --> src/main.rs:67:32
   |
67 |         for element in v.split_whitespace().collect::<Vec<&str>>() {
   |                        ^ borrowed value does not live long enough
...
70 |         main_vector.push(sub_vector);
   |         ----- borrow later used here
71 |     },
   |     - `v` dropped here while still borrowed
```

I think it's about the references and borrowing but still I'm having hard time to figure this out.

[reference](#) [rust](#) [lifetime](#) [borrowing](#)

Share

[Improve this question](#)

[Follow](#)

[edited Nov 24 '19 at 21:45](#)

[asked Nov 24 '19 at 20:45](#)



orhun

49 • 1 • 5

[Add a comment](#)

2 Answers

[Active](#) [Oldest](#) [Votes](#)



2



This question is similar to [Return local String as a slice \(&str\)](#). And the easiest solution is the same as in that question - **use String, not &str**. These questions are different as that answer specifically talks about returning from a function, and you have no function listed.

**Your privacy** To address why lifetimes make your code fail, try a simpler example

By clicking "Accept all cookies", you agree Stack Exchange can store cookies on your device and disclose information in accordance with our [Cookie Policy](#).

[Accept all cookies](#)

[Customize settings](#)

```
fn main() {
    let mut v:Vec<&str> = Vec::new();
    {
        let chars = [b'x', b'y', b'z'];
        let s:&str = std::str::from_utf8(&chars).unwrap();
        v.push(&s);
    }
    println!("{}", v);
}
```

and compiler output

```
let s:&str = std::str::from_utf8(&chars).unwrap();
      ^^^^^^ borrowed value does not live long enough
```

The reason this doesn't work is exactly what the compiler says. `chars` is created inside the block, so it gets a lifetime associated with that block, and when your program exits that block, `chars` might not exist anymore. Anything that referred to `chars` might have a dangling pointer. Rust avoids dangling pointers by making this illegal. In my example it seems silly that Rust doesn't allow this, but in yours it makes sense - Rust can keep the stack small by deleting the old `str s` from `v.split_whitespace().collect::<Vec<&str>>()` every time through the loop.

[Share](#)

[Improve this answer](#)

Follow

answered Nov 24 '19 at 21:57



[asky](#)

1,100 • 10 • 20

It's solved by using `Vec<Vec<String>>` instead of `Vec<Vec<&str>>`. Thank you!

– [orhun](#)

Nov 24 '19 at 22:07

Thanks @orhun for letting us know about `String` working (where `&str` didn't). This solved my problem as well.

– [Jake Ireland](#)

Oct 11 at 1:26

[Add a comment](#)



0



It would be better if more code was provided, but I tried to reproduce it and arrived at this working snippet:

```
type Error = i32;

struct Reader
{
    lines: Vec<Result<String, Error>>
}

impl Reader
{
    pub fn new() -> Self
    {
        Self{lines: vec![Ok("foo".into()), Ok("bar".into())]}
    }

    pub fn lines(&self) -> &[Result<String, Error>]
    {
        &self.lines
    }
}

fn main() {
    let reader = Reader::new();
    let mut main_vector: Vec<Vec<&str>> = Vec::new();
    for line in reader.lines() {
        match line {
            Ok(v) => {
                let mut sub_vector: Vec<&str> = Vec::new();
                for element in v.split_whitespace().collect::<Vec<&str>>() {
                    sub_vector.push(element);
                }
                main_vector.push(sub_vector);
            },
            Err(e) => panic!("failed to parse: {:?}", e),
        }
    }
}
```

You can check it on Rust Playground here <https://play.rust-lang.org/?version=stable&mode=debug&edition=2018&gist=f2785fcad682b9dd1f5ed61c7e0308d8>

[Share](#)

[Improve this answer](#)

Follow

answered Nov 24 '19 at 21:34



[Fazer](#)

32 • 5

I've added other parts of the code. This snippet works well but I think the `BufReader ( .lines() )` is the problem.

– [orhun](#)

[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 [reference](#) [rust](#) [lifetime](#) [borrowing](#) 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

0

[Programatically create slice of string slices on Rust](#)

[Return local String as a slice \(&str\)](#)

#### Related

[The located assembly's manifest definition does not match the assembly reference](#)

["borrowed value does not live long enough" when using the builder pattern](#)

[Factory method: instance does not live long enough](#)

[Error: borrowed value is only... reference must be valid for](#)

[Value does not live long enough](#)

3

[Value does not live long enough with a generic function that creates a container, adds items to it, then iterates over the items](#)

[Problems with Tuple's lifetime in rust.](#)

2

[Can't borrow reference to structure in captured tree because it doesn't live long enough](#)






1

[Borrowed value does not live long enough with Arc, thread and channel](#)

1

[Nesting Structs: "borrowed value does not live long enough"](#)

#### Hot Network Questions

-  [How does a river freeze when the water keeps moving?](#)
-  [What's the social meaning of "He was a student of..."?](#)
-  [what is the official procedure if you answer "yes" to any of the US visa \(DS 160\) "are you a terrorist/slaver/bodysnatcher" questions?](#)
-  [What does this entry on the Rocinante's pilot quick-menu mean?](#)
-  [Numbers, Racked Up](#)

[more hot questions](#)

 [Question feed](#)

[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](#)

site design / logo © 2021 Stack Exchange Inc; user contributions licensed under [cc by-sa](#). rev 2021.12.22.41046