



Products

# How can I split strings without worrying about mutable borrows?

[Log in](#) or [sign up](#)

[Ask Question](#)

Asked 1 year ago

Active 1 year ago

Viewed 345 times



1



I'm currently trying to solve [this challenge](#) to learn Rust. I've created my own solution below:

```
use std::io;
use std::vec::Vec;
use std::collections::HashSet;

fn main() {
    let stdin = io::stdin();
    let mut buffer = String::new();
    let mut set = HashSet::new();
    let req = ["byr", "iyr", "eyr", "hgt", "hcl", "ekl", "pid", "cid"];
    let mut res: u32 = 0;

    while stdin.read_line(&mut buffer).unwrap() > 0 {
        if buffer == "\n" {
            for r in req.iter() {
                if set.contains(r) {
                    set.remove(r);
                }
            }

            if (set.len() == req.len()) || (set.len() == 7 && !set.contains("cid")) {
                res += 1;
            }
        } else {
            if buffer.ends_with('\n') {
                buffer.pop();
            }

            let inputs: Vec<&str> = buffer.split_whitespace().collect();

            for key_val in inputs {
                let kval: Vec<&str> = key_val.split(':').collect();

                set.insert(kval[0]);
            }
        }
    }
}
```

When I tried to compile it, the compiler outputs the error:

```
error[E0502]: cannot borrow 'buffer' as mutable because it is also borrowed as immutable

if (set.len() == req.len()) || (set.len() == 7 && !set.contains("cid")) {
|
|   --- immutable borrow later used here
...
28 |     let inputs: Vec<&str> = buffer.split_whitespace().collect();
|                               ----- immutable borrow occurs here
...
37 |     buffer.clear();
|     ~~~~~^ mutable borrow occurs here
```

I've solved other problems by using this kind of style before (using `&mut buffer` and `split` it somewhere) without getting this error.

Why does this error occur here and how can I fix it?

[rust](#)

[Share](#)

[Improve this question](#)

[Follow](#)

edited Dec 8 '20 at 14:29



Shepmaster

305k ● 59 ● 824 ● 1083

asked Dec 8 '20 at 13:51



Christopher Namchee

60 ● 1 ● 8

Did you already read this? [doc.rust-lang.org/book/ch04-02-references-and-borrowing.html](https://doc.rust-lang.org/book/ch04-02-references-and-borrowing.html)

– mKrieger1

Your privacy Dec 8 '20 at 13:54

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 | Why buffer.split\_whitespace is considered as immutable borrow

– Christopher Namchee

Dec 8 '20 at 14:00

It is an immutable borrow because it returns slices borrowed from the original `buffer`. As long as they live, it is considered borrowed.

– [rodrigo](#)

Dec 8 '20 at 16:19

@rodrigo, but I have declared `buffer` as mutable, shouldn't the borrow considered to be mutable as well?

– [CristopherNamchee](#)

Dec 8 '20 at 23:37

1

No, `split_whitespace()` takes a `&self`, not a `&mut self`, so it borrows the buffer immutably, it does not matter if it is declared `mut` or not. But even if it were mutably borrowed, that would not help your code, because you cannot mutably borrow a value twice.

– [rodrigo](#)

Dec 8 '20 at 23:48

[Add a comment](#)

1 Answer

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



3



Look at the type of your `set` variable. You can get it by writing a `set` as `()` just after the `set.insert(...)` line. You'll see this error message:

```
non-primitive cast: 'HashSet<&str>' as '()'
```

So your `set` holds references to `str`. But where do these references live. Well, all your `str` ultimately come from `buffer` so they are references to the contents of `buffer`. If you were allowed to do `buffer.clear()` all these references would become invalidated.

You have two basic options:

1. Keep all your lines in memory all the time so that your references in `set` are not invalidated. You could use ``read``
2. Store `String` instead of `&str` values in the `set`.

The easier solution is #2:

```
set.insert(kval[0].to_owned());
```

With that and a few other minor fixes, your algorithm will compile ([playground](#)).

Option #1 would require to read the whole `stdin` into a variable and then iterate over the lines:

```
use std::io::Read;
let mut stdin = io::stdin();
let mut input = String::new();
stdin.read_to_string(&mut input).unwrap();
for buffer in input.lines() {
    ...
}
```

Now `buffer` is a `&str` instead of a `String`, but the lifetime of the values in `set` will be that of `input` that is outside of the loop. You would need a few minor fixes, particularly with the end-of-line checks.

[Share](#)

[Improve this answer](#)

[Follow](#)

[edited Dec 8 '20 at 14:13](#)

[answered Dec 8 '20 at 14:07](#)



[rodrigo](#)

**83.3k** ● 10 ● 129 ● 170

Thanks, but why `String` though? Can I use `&str` instead?

– [CristopherNamchee](#)

Dec 8 '20 at 23:53

@CristopherNamchee: Do you mean as a type in the `set`? Because `&str` is a borrowed type, by definition, because it is a reference, while a `String` is not. In order for a (non-`'static`) `&str` to exist there must be a borrowed value somewhere.

– [rodrigo](#)

Dec 9 '20 at 0:29

[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](#) 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

Related

How do I split a string in Rust?

215  
How do I create a global, mutable singleton?

How do I concatenate strings?

4  
How can callbacks with captured mutable variables be treated like normal mutable borrows?






Borrow errors for multiple borrows

How to deal with cyclic mutable borrows

Mutable vs. immutable borrows in closure?

Recursive async function that borrows mutable variable twice

Hot Network Questions

-  Why did the JWST solar array deploy early?
  -  Is there a deterministic guide to landing?
  -  Seeing oneself in an abstract painting
  -  I've got a material setup that blends two shaders decently, but how would I apply it to a circular target?
  -  Without passport stamps, can my country's authorities still know what countries I've visited?
- [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  
Privacy 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