

Lifetime problem when passing data to external crate

[help](#)

[CoLDSToRAGE](#) February 26, 2019, 5:12pm 1

I'm a total newbie trying to learn RUST by developing a 'faux' BBC Micro Emulator...

Buy 'faux' I mean that I'm not emulating the hardware as such, but simply developing a BASIC Interpreter that emulates the end result. At present my code can display a screen, render text, accept keyboard input, store basic programs, plays sound and can already cater for some rudimentary commands such as NEW, CLS, LIST and so on.

The time came for me to get into the nitty-gritty of language parsing. So I decided to make use of Dmitry Soshnikov's LALR(1) parser, to which I am passing values by having it declared as an external crate, then a declared use, and then priming it using `let mut parser = Parser::new();`

In my error below, if I replace `&_parameter_line_with_spaces` with `"2 + 2"` then I get '4' as expected. But when I try to use a referenced variable as below (a String containing '2 + 2'), then I get the borrowing error.

```
error[E0597]: `_parameter_line_with_spaces` does not live long enough
--> src\main.rs:540:58
|
540 | let _result = parser.parse (&_parameter_line_with_spaces);
| ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^ borrowed value does not live long enough
...
577 | }
| - borrowed value only lives until here
|
= note: borrowed value must be valid for the static lifetime...
```

I'm old enough to say that I cut my teeth on BBC BASIC back in the day, and then moved on to Pascal, Cobol, Natural, PHP and Javascript... but no C++. I thought I'd dive with a RUST project that would hold my attention. I have fought with the concept of lifetimes and borrowing, and although I grasp the principles involved and I've read the Klabnik & Nichols book, I still stumble.

All comments gratefully received.

[RustyYato](#) February 26, 2019, 5:31pm 2

Could you make a minimal viable example?

There isn't enough info to give any help.

- how is the `Parser` type defined?
- how is `Parser::parse` defined?
- what is the context around line 540?
- etc.

Once you get a minimal viable example, could you please post a [playground link](#). To do so, go to the link and put your code in the editor and click the share button in the top right corner. Copy the permalink and post that here. Note that playground has very limited support for external crates so if you can't use playground, at least post the definitions of the relevant types and functions here.

[vitalyd](#) February 27, 2019, 2:00am 3

I had to google for this parser but the issue is <https://github.com/DmitrySoshnikov/syntax/blob/master/src/plugins/rust/templates/lr.template.rs#L177> wants a `&'static str`. I'm not sure why that would be the requirement, but I stopped looking . This is why `"2 + 2"` literal works but not when you start using a local `String`.

[CoLDSToRAGE](#) March 4, 2019, 11:24am 4

Hey Krishna, thanks for taking the time to reply...

Vitalyd has hit the nail squarely on the head, and has provided the perfect answer for me.

[CoLDSToRAGE](#) March 4, 2019, 11:28am 5

Perfect!

Thanks so much for taking the time to answer, and indeed supply the correct solution for me.

For anyone else who needs to action such a solution, I found the following code that converts a `String` to a `&'static str`. It is indeed odd to require a parameter passed in this way, from what I can tell from other members of the Rust community, the better way is to use a crossbeam scope (of which I have no knowledge at the time of writing).

```
fn string_to_static_str(s: String) -> &'static str {
Box::leak(s.into_boxed_str())
}
```

Source: [rust - How to convert a String into a &'static str - Stack Overflow](#)

Related topics

Topic	Replies	Views	Activity
Borrowed value does not live long enough to be valid for the static lifetime expected help	6	2371	January 12, 2023
Preventing legacy mutable variable borrowing to solve lifetime problem help	5	250	June 19, 2024
Borrowed value does not live long enough? help	6	5791	January 12, 2023
Variable Lifetimes help	4	369	August 30, 2023
Issues with self-borrowing methods w/ lifetime parameters help	3	616	May 21, 2020

- [Home](#)
- [Categories](#)
- [Guidelines](#)
- [Terms of Service](#)