

[Skip to main content](#)

[Stack Overflow](#)

1. [About](#)

2. [Products](#)

3. [For Teams](#)

1. [Stack Internal](#) Implement a knowledge platform layer to power your enterprise and AI tools.

2. [Stack Data Licensing](#) Get access to top-class technical expertise with trusted & attributed content.

3. [Stack Ads](#) Connect your brand to the world's most trusted technologist communities.

4. [Releases](#) Keep up-to-date on features we add to Stack Overflow and Stack Internal.

5. [About the company](#) [Visit the blog](#)

Loading...

[current community](#)

[Stack Overflow](#)

[help](#) [chat](#)

- [Meta Stack Overflow](#)

your communities

[Sign up](#) or [log in](#) to customize your list.

[more stack exchange communities](#)

[company blog](#)

3. [Log in](#)

4. [Sign up](#)

[AI Assist is now on Stack Overflow](#). Start a chat to get instant answers from across the network. Sign up to save and share your chats.

[Home](#)

[Questions](#)

[AI Assist](#)

[Tags](#)

[Challenges](#)

7. [Chat](#)

[Articles](#)

[Users](#)

[Companies](#)

[Collectives](#)

Communities for your favorite technologies. [Explore all Collectives](#)

Stack Internal

Stack Overflow for Teams is now called **Stack Internal**. Bring the best of human thought and AI automation together at your work.

[Try for free](#) [Learn more](#)

[Stack Internal](#)

Bring the best of human thought and AI automation together at your work. [Learn more](#)

Collectives™ on Stack Overflow

Find centralized, trusted content and collaborate around the technologies you use most.

[Learn more about Collectives](#)

Stack Internal

Knowledge at work

Bring the best of human thought and AI automation together at your work.

[Explore Stack Internal](#)

[Rust "borrowed value does not live long enough" disappears when I remove a piece of code](#)

[Ask Question](#)

Asked 1 year, 10 months ago

Modified [1 year, 10 months ago](#)

Viewed 100 times

0

I am using the solution in this question:

<https://stackoverflow.com/a/56925633/1634905>

My code looks like:

```
fn get_room_header<'a>(socket: &'a SocketRef) -> Option<&'a str> {
    socket.req_parts().headers.get("room")?.to_str().ok()
}

async fn on_connect(socket: SocketRef) {

    println!("on_connect");

    let Some(room) = get_room_header(&socket) else {
        println!("No room sent: {:?}",socket.req_parts());
        _ = socket.disconnect();
        return;
    };

    // println!("Room: {:?}",room);

    let _ = socket.leave_all();
    let _ = socket.join(room);//if I comment out this line, it no longer throws error

}
```

This doesn't compile and gives an error:

```
`socket` does not live long enough
borrowed value does not live long enough
main.rs(61, 1): `socket` dropped here while still borrowed
main.rs(46, 21): binding `socket` declared here
main.rs(50, 22): argument requires that `socket` is borrowed for `'static`
```

Screenshot of error:

If I comment out the line `let _ = socket.join(room);`, then it no longer throws the error. Why's this?

The `join()` function is from `rust_socketio` crate:

<https://github.com/1c3t3a/rust-socketio/tree/main>

It looks like this:

```
pub fn join(&self, rooms: impl RoomParam) -> Result<(), A::Error> {
    self.ns.adapter.add_all(self.id, rooms)
}

pub trait RoomParam: 'static {
    /// The type of the iterator returned by `into_room_iter`.
    type IntoIter: Iterator<Item = Room>;

    /// Convert `self` into an iterator of rooms.
    fn into_room_iter(self) -> Self::IntoIter;
}
```

• [rust](#)

[Share](#)

[Improve this question](#)

Follow

[edited Jan 30, 2024 at 3:08](#)

[asked Jan 30, 2024 at 2:10](#)

2

There isn't quite enough information to answer the question; *at a minimum* we need to see the signature of the `join()` function and of its containing `impl` block, but Stack Overflow strongly prefers *reproducible* code. Please try to condense your situation into code we could actually try compiling to reproduce the error.

Kevin Reid – [Kevin Reid](#)

2024-01-30 02:54:43 +00:00

Commented Jan 30, 2024 at 2:54

@KevinReid I have added the `join()` function to my post. It's from `socketioxide` crate.

sudoExclamationExclamation – [sudoExclamationExclamation](#)

2024-01-30 03:09:09 +00:00

Commented Jan 30, 2024 at 3:09

[Add a comment](#) | .

1 Answer

Sorted by: [Reset to default](#)

Highest score (default) Trending (recent votes count more) Date modified (newest first) Date created (oldest first)

4

```
pub fn join(&self, rooms: impl RoomParam) -> Result<(), A::Error> {
```

This says that whatever you pass to `join()` must be of a type that implements `RoomParam`.

```
pub trait RoomParam: 'static {
```

This says that one of the requirements of implementing `RoomParam` is that it is `'static`. Since you passed a `&'a str`, the compiler concludes that `'a = 'static`. This then requires, through the signature of `get_room_header()`,

```
fn get_room_header<'a>(socket: &'a SocketRef) -> Option<&'a str> {
```

that `socket` also be borrowed for `'static`. But it is not, and in fact you drop it at the end of the function, so you get a “borrowed value does not live long enough” error.

Now as to solving the problem, the general thing to keep in mind is: if you get an error that something must be borrowed for `'static`, that most often means that *it shouldn't be borrowed at all*. Don't try to use a `&str` as the `RoomParam` — use an owned value.

If we go to [the documentation for RoomParam](#), we can see it is implemented for several types and you should probably use `String` — that is, convert your `&str` to `String`:

```
let _ = socket.join(room.to_owned());
```

[Share](#)

[Improve this answer](#)

Follow

answered Jan 30, 2024 at 3:24

[Kevin Reid](#)

Sign up to request clarification or add additional context in comments.

1 Comment

Add a comment

sudoExclamationExclamation

[sudoExclamationExclamation](#) Over a year ago

Thanks. That worked. Your answer helped me change my function to return `Option<String>` instead of the original `Option<&'a str>`. So now, my function is: `fn get_room_header(socket: &SocketRef) -> Option<String> { socket.req_parts().headers.get("room")?.to_str().ok().map(|s| s.to_string()) }`

2024-01-30T04:09:20.087Z+00:00

0

Reply

- Copy link

Your Answer

Thanks for contributing an answer to Stack Overflow!

- Please be sure to *answer the question*. Provide details and share your research!

But *avoid* ...

- Asking for help, clarification, or responding to other answers.
- Making statements based on opinion; back them up with references or personal experience.

To learn more, see our [tips on writing great answers](#).

Sign up or [log in](#)

Sign up using Google

Sign up using Email and Password

Submit

Post as a guest

Name

Email

Required, but never shown

Post as a guest

Name

Email

Required, but never shown

Post Your Answer Discard

By clicking "Post Your Answer", you agree to our [terms of service](#) and acknowledge you have read our [privacy policy](#).

Start asking to get answers

Find the answer to your question by asking.

[Ask question](#)

Explore related questions

- [rust](#)

See similar questions with these tags.

- The Overflow Blog
[Introducing Stack Overflow AI Assist—a tool for the modern developer](#)
[Treating your agents like microservices](#)

- [Featured on Meta](#)

[Chat room owners can now establish room guidelines](#)

[AI Assist is now available on Stack Overflow](#)

[Policy: Generative AI \(e.g., ChatGPT\) is banned](#)

Linked

[17](#)

[Is there simpler method to get the string value of an Actix-Web HTTP header?](#)

Related

[422](#)

[Why does the Rust compiler not optimize code assuming that two mutable references cannot alias?](#)

[0](#)

[Rust: Borrowed value does not live long enough](#)

[0](#)

[Rust E0597 - borrowed value does not live long enough](#)

[0](#)

[Rust loop - borrowed value does not live long enough](#)

[0](#)

[borrowed value does not live long enough in Rust](#)

[0](#)

[Rust reporting "borrowed value does not live long enough" incorrectly?](#)

Hot Network Questions

- [What international legal mechanisms are in place to stop a country from outright expropriating foreign assets?](#)
- [2 ways of resolution of vectors](#)
- [Misunderstanding the lorentz transformation](#)
- [Do something and "win"](#)
- [A very weird integral equation](#)
- [Preservation of universal sentences under substructures in higher-order logic](#)
- [TikZ word search diagram v2](#)
- [Why does Mtt 13:34 say that Jesus always spoke in parables?](#)
- [When did "doubt" meaning question or point of uncertainty fall out of use outside of Indian English?](#)
- [Proportional odds logistic regression for ordered category outcome - how to convert odds ratios to probabilities in this case?](#)
- [A hilarious parody of Western movies where an entire town collapses into tunnels](#)
- [Why is "cmd | xargs" to join lines of cmd with spaces wrong](#)
- [How to protect 100-year-old wooden floors for the next 100 years?](#)
- [How can I secure a loose kitchen cabinet?](#)
- ["Save as webpage, HTML Only" Fail while offline, Firefox](#)
- [Is zero-point energy a mathematical artifact?](#)
- [Why does black allow white to have a protected passed central pawn here?](#)
- [Bone as a structural material; or, making a building's skeleton literal](#)
- [Finding real variable where a 2x2 complex matrix is singular in Mathematica](#)
- [Correct website for Thailand ETA application form](#)
- [Is Codepage 1252 guaranteed to be available on all Windows systems?](#)
- [Is "We will review application starting from..." a deadline?](#)
- [Download is performed unsandboxed as root - Why am I getting this message, and how do I fix it?](#)
- ["Us" in Hegel's "Greater Logic"](#)

[more hot questions](#)

[Question feed](#)

Subscribe to RSS

Question feed

To subscribe to this RSS feed, copy and paste this URL into your RSS reader.

[Stack Overflow](#)

- [Questions](#)
- [Help](#)
- [Chat](#)

[Business](#)

- [Stack Internal](#)
- [Stack Data Licensing](#)
- [Stack Ads](#)

[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](#)