Rust lifetimes with closures using hyper

Ask Question

Asked 1 year, 6 months ago
Active 1 year, 6 months ago
Viewed 202 times

1

Been learning rust and having a problem with lifetime when passing `conn` to the `request_handler`. I get an error saying

```
error[E0312]: lifetime of reference outlives lifetime of borrowed content...
  --> src/main.rs:33:70
   |
33 |         let request_handler = |req: Request<Body>| async { request_handler(conn, req).await };
   |                                                                      ^^^^
   |
   = note: ...the reference is valid for the static lifetime...
```

but I am not sure how to handle lifetimes with closures and why/how it is static. I have a loose understanding of lifetimes and borrowing but this seems like a more complex case. I also would just not use a closure, but the return type of one of the closures has a type that is not exported by the hyper crate, so i don't know how i would create a `fn` without being able to declare the return type.

Also I can confirm if i remove passing `conn` i can get everything to work, but I want to use the `conn` object in the `request_handler`.

```
use hyper::server::conn::AddrStream;
use hyper::service::make_service_fn;
use hyper::Version;
use hyper::{Body, Error, Method, Request, Response, Server};
use std::net::{IpAddr, Ipv4Addr, SocketAddr};
mod http_models;
mod utils;
use hyper::service::service_fn;

async fn request_handler(
    conn: &'static AddrStream,
    req: Request<Body>,
) -> Result<Response<Body>, hyper::Error> {
    println!("req: {:?}", req);
    if req.method() == Method::CONNECT {
        println!("Connect")
    }
    let res: Response<Body> = Response::builder()
        .status(200)
        .version(Version::HTTP_11)
        .body(Body::empty())
        .unwrap();
    return Ok(res);
}

#[tokio::main]
async fn main() -> Result<(), Error> {
    let ip = IpAddr::V4(Ipv4Addr::new(127, 0, 0, 1));
    let addr = SocketAddr::new(ip, 1337);
    //let client = Client::new();

    let make_service = make_service_fn(|conn: &AddrStream| async {
        let request_handler = |req: Request<Body>| async { request_handler(conn, req).await };
        let service = service_fn(request_handler);
        Ok::<_, Error>(service)
```

rust lifetime hyper

Share
Improve this question
Follow

asked Jun 18 '20 at 21:48

Gekctek
1,099 ● 2 ● 10 ● 23

You do not use the `conn` in `request_handler`, if you remove this argument the problem just disappears. Can you do that or your real code is more complex?
– rodrigo
Jun 18 '20 at 23:45

I will be using it even though it's not being used right now
– Gekctek
Jun 19 '20 at 0:05

Add a comment

1 Answer

Active Oldest Votes

2

The `conn` argument of the closure passed to `make_service_fn` only lives as long as the closure body, but the return value of the closure (`Ok(service)`) references it. The closure must have the type `FnMut(&Target) -> impl Future` , which means it sadly is not permitted to return a value that references its argument.

The only solution is to copy/clone whatever you need from `conn` while setting up your request handler, since you cannot keep a reference to it once the closure returns.

Share
Improve this answer
Follow

answered Jun 19 '20 at 4:57

Coder-256
4,394 ● 1 ● 19 ● 47

Cool. That helps a lot ty. I saw other examples of items being cloned and didn't fully understand their purpose
– Gekctek
Jun 19 '20 at 15:22

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 lifetime hyper or ask your own question.

Related

Lifetimes in Rust

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

Why are explicit lifetimes needed in Rust?

Taking closures with explicit lifetimes as arguments in Rust

2
Creating a hyper service with custom error type

hyper client cannot lookup address information for server running on IPv6 localhost

4
Passing additional state to rust `hyper::service::service_fn`

Understanding lifetimes: max lifetime and 'static

Hot Network Questions

- Is it acceptable to omit "about" in this sentence? "I love everything (about) math."
- What caused this crash landing?
- Naruto fighting game with Hulk and Homer Simpson?
- How much of the English history in this Decameron story has any basis in fact?
- Tax implications of large gift

more hot questions

Question feed