Implementing Game Server In Rust

help

shekohex October 29, 2018, 10:48pm 1

Hello There,

I love working with Network things specially for game development , for a long time i had played a pretty old MMORPG game that was released in 2003.
I found the old Game client somewhere on the internet and i also found a working server emulator for that game written in C# but it's modern and using Asynchronous programming Which i loved it.
But as i said i'm trying to improve my skills in Rust and also in Asynchronous programming.
So after some time i got how the game server works and how the communication between Server and client is done.

The game server is sending and receiving packets over a tcp connection, and it uses a structured data with **prefixed length** of type `u16`
The structure is really simple

```
Packet Length: u16,
Packet ID: u16,
Packet Body: Vec<u8>,
```

And of course the length of the Packet body is equal to packet length - 4 Bytes.

And as all of us knows that tcp is a strem protocol and it doesn't have mechanism to handle Packet fragments (slices) so you need to implement that at the application layer.

The way that provided in C# is that after receiving the connection it creates an Async Task (aka: Future) and in that task it reads the packet length form the buffer (stream) and then get a slice of that buffer equal to that length and it makes a new task to process that packet and sends the result (response) back To the socket (game client).
That way it makes sure that it process every segment individually without overlapping.

I'm trying to figure out how can i implement such a thing in Rust.

I'm using `tokio` as a Runtime also I'm using `bytes` and `bytesorder` crate to deal with Bytes.

I already have read the docs of tokio-rs and also the chat example seems interesting..

If someone could guide me with a simple example to make a simple tcp server that receiving a `prefixed length` messages and process each message individually in an Asynchronous way that will be great and very helpful.

I have to mention that it's just for learning purpose.

vitalyd October 30, 2018, 2:56am 2

You'll want to look at the length_delimited module in tokio - it has a codec that handles length prefixed payloads.

As for an example, here's a modified (read: stripped down) version from https://github.com/tokio-rs/tokio/blob/master/examples/print_each_packet.rs:

```
extern crate tokio;

use tokio::net::TcpListener;
use tokio::prelude::*;
use tokio::codec::length_delimited::Builder;

use std::env;
use std::net::SocketAddr;

fn main() {
   let addr = env::args().nth(1).unwrap_or("127.0.0.1:8080".to_string());
   let addr = addr.parse::<SocketAddr>().unwrap();

   let socket = TcpListener::bind(&addr).unwrap();
   println!("Listening on: {}", addr);

   let done = socket
       .incoming()
       .map_err(|e| println!("failed to accept socket; error = {:?}", e))
       .for_each(move |socket| {
           // Define the length delimited codec here - there are more options, so read
           // the module's docs
           let framed = Builder::new().length_field_length(std::mem::size_of::<u16>()).new_framed(socket);
           // split the socket into (framed) read and write portions
           let (writer, reader) = framed.split();

           let proc = reader
                // in this closure is where you'd perform your frame/request handling
                // here I'm just echoing back the buffer
               .and_then(|frame| Ok(frame.freeze()))
               .map_err(|_| ())
                // we forward() all responses back to the client
                // the same codec will now take Bytes from us, and encode (serialize) them
                // to the client
               .forward(writer.sink_map_err(|_| ()))
               .map(|_| ());

           // Now we spawn the above state machine onto the reactor - each client (accepted tcp conn)
           // will run the state machine independent of all others
           tokio::spawn(proc)
       });

   tokio::run(done);
}
```

I've omitted error handling (except mapping them away), and I'm not sure what's involved in your request handling. But the above is a rough sketch of what your server might look like.

6 Likes

shekohex October 30, 2018, 6:39pm 3

@vitalyd thank you very much for you help.

I just forget to mention that the game client is sending an encrypted packet, but that is not the problem, i already know the encryption algorithm and i done that, the problem it that the header of the packet which is the length and the packet id is also encrypted, so this Codec will read a wrong length, is there a way to process the first 2 bytes of the packet (*size_of::u16*) so that it can read the right length, or should i copy/implement that codec with that modification ?

> Note: I already had a quick look at the Codec API and it is really good an it have good examples too, but i can't find a way that cover that case

shekohex October 30, 2018, 9:22pm 4

shekohex:

> should i copy/implement that codec with that modification

I did it and it works
thank you @vitalyd again for your help <3

coco March 7, 2019, 5:30am 5

I guess you were talking about Maplestory.

1 Like

Related topics

Topic	Replies	Views	Activity
Best way to send Rust struct over Async TcpStream? help	6	3849	March 25, 2021
Write_all and read using framed ... Not working Sending file over the network help	7	1402	July 16, 2021
Best way to send Frame over Tcp Stream? help	4	1702	February 2, 2023
Wrapping my head around tokio server/client help	6	972	January 12, 2023
I need deep help to deeply understand the code in simple language help	5	195	September 19, 2025