What does "borrowed data cannot be stored outside of its closure" mean?

Asked 3 years, 6 months ago

Active 11 months ago

Viewed 2k times

▲

7

▼ 🔖 🕘

When compiling the following code:

```
fn main() {
    let mut fields = Vec::new();
    let pusher = &mut |a: &str| {
        fields.push(a);
    };
}
```

The compiler gives me the following error:

```
error: borrowed data cannot be stored outside of its closure
 --> src/main.rs:4:21
  |
3 |     let pusher = &mut |a: &str| {
  |                  ------   --------- ...because it cannot outlive this closure
  |                  |
  |                  borrowed data cannot be stored into here...
4 |         fields.push(a);
  |                     ^ cannot be stored outside of its closure
```

And in later versions of Rust:

```
error[E0521]: borrowed data escapes outside of closure
 --> src/main.rs:4:9
  |
2 |     let mut fields = Vec::new();
  |         --------- `fields` declared here, outside of the closure body
3 |     let pusher = &mut |a: &str| {
  |                        - `a` is a reference that is only valid in the closure body
4 |         fields.push(a);
  |         ^^^^^^^^^^^^^^^ `a` escapes the closure body here
```

What does this error mean, and how can I fix my code?

`rust`   `closures`   `borrow-checker`   `ownership`

Share
Improve this
question
Follow

edited Jan 5 at 18:36

Shepmaster
305k ● 59 ● 824 ● 1083

asked Jun 30 '18 at 19:48

Valentin Lorentz
9,127 ● 5 ● 45 ● 64

Add a comment

1 Answer

Active   Oldest   Votes

▲

6

▼

✔

🕘

It means exactly what it says: that the data you are borrowing only lives for the duration of the closure. Attempting to store it outside of the closure would expose the code to memory unsafety.

This arises because the inferred lifetime of the closure's argument has no relation to the lifetimes stored in the `Vec`.

Generally, this isn't a problem. If you were *not* using this closure, the usual and more inference to happen. In this case, you can add a type to `fields` and remove it from the closure:

```
let mut fields: Vec<&str> = Vec::new();
let pusher = |a| fields.push(a);
```

answered Jun 30 '18 at 20:36

Shepmaster
305k ● 59 ● 824 ● 1083

Oh, thank you! I would have expected an error telling me the borrowchk does not have enough info.
– Valentin Lorentz
Jun 30 '18 at 22:40

@ValentinLorentz FWIW, I've never seen this particular error before, and this is the first SO question about it. It's possible this is a brand new error. You may wish to file a bug with your case and the error message and say that it wasn't helpful. The Rust contributors strive to have useful error messages.
– Shepmaster
Jul 1 '18 at 3:36

I spent a few hours misinterpreting the "and remove it from the closure" part. It's not that you can remove it if you're bored with it, you *have to* remove it to make this compile.
– Stein
Aug 8 '20 at 19:34

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 closures borrow-checker ownership 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

Linked

Store a vec of references to objects in another data structure

1
borrowed data cannot be stored outside of its closure

Related

In PHP, what is a closure and why does it use the "use" identifier?

How to include and use Rust libraries

No method parse for String

Cannot move out of borrowed content when trying to transfer ownership

Unable to coerce &String to &str

Factory method: instance does not live long enough

How to convert C variable-length array code to Rust?

0

Pass a Struct<'a>::method as a `for<'a> Fn(&Foo<'a>)` for use in a closure

How to use delayed initialization in rust code and pass compiler "possibly-uninitialized variable" rule?

Hot Network Questions

How much of the English history in this Decameron story has any basis in fact?

which one of these paths has the priority: /usr or /usr/local

Why does the prophecy imply Macbeth has to murder the king?

What does the numbers mean in this guitar tab if they already gave the chords to play?

Text overflow in tabularx

more hot questions

Question feed