



Products

## Why does the closure take ownership of the vector here?

Login Sign up

Ask Question

Asked 4 years, 10 months ago

Active 4 years, 10 months ago

Viewed 144 times



4



The rust documentation has this example in the [Closures](#) section.

```
let nums = vec![1, 2, 3];  
  
let takes_nums = || nums;  
  
println!("{}", nums);
```

The documentation says

If your closure requires it, however, Rust will take ownership and move the environment instead

And the above code results in this error

```
note: `nums` moved into closure environment here because it has type  
[closure()] -> collections::vec::Vec<i32>], which is non-copyable  
let takes_nums = || nums;  
                  ~~~~~
```

for which the docs say

Vec has ownership over its contents, and therefore, when we refer to it in our closure, we have to take ownership of nums. It's the same as if we'd passed nums to a function that took ownership of it.

I don't understand why the closure doesn't just borrow the ownership of the vector as it does in this example from the documentation

```
let num = 5;  
let plus_num = |x: i32| x + num;  
  
assert_eq!(10, plus_num(5));
```

This closure, plus\_num, refers to a let binding in its scope: num. More specifically, it borrows the binding.

rust

Share

Improve this question

Follow

asked Mar 1 '17 at 14:49



SphericalCow

345 ● 1 ● 2 ● 13

Add a comment

1 Answer

Active Oldest Votes



5



The answer lies in the signature of the closure: what does takes\_num return?

It returns nums, whose type is Vec<i32>.

To give ownership of something to someone, you must first own it, otherwise it's not yours to give. The same rule applies to the closure.

Share

Improve this answer

Follow

By clicking "Accept all cookies", you agree Stack Exchange can store cookies on your device and disclose information in accordance with our [Cookie Policy](#).

answered Mar 1 '17 at 14:54

Accept all cookies Customize settings



Matthieu M.

261k ● 40 ● 396 ● 665

[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](#) 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

#### Related

[What are move semantics in Rust?](#)

[Returning closures which capture outer variables in Rust](#)

[Why does a closure introduce a borrow when the same code inlined does not?](#)

[Why is the string literal not moved?](#)

6

[How can one force Rust to take ownership of memory allocated other than by its safe methods?](#)

1

[Rust Closures concept](#)

#### Hot Network Questions

 [What does the numbers mean in this guitar tab if they already give the chords to play?](#)

 [Schrödinger's cat program](#)

 [Does anyone know what this fan blower thingy is?](#)

 [Does saying "Keep it up" put me in an authoritative position?](#)

 [Remove the Times x from display in Inactivate expressions in V13](#)

[more hot questions](#)

 [Question feed](#)

#### STACK OVERFLOW

[Questions](#)  
[Jobs](#)  
[Developer Jobs Directory](#)  
[Salary Calculator](#)  
[Help](#)  
[Mobile](#)

#### PRODUCTS

[Teams](#)  
[Talent](#)  
[Advertising](#)  
[Enterprise](#)

#### 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](#)

site design / logo © 2021 Stack Exchange Inc; user contributions licensed under [cc by-sa](#). rev 2021.12.22.41046