Annotate function to allow mutable borrows of disjoint fields

help

mrborrowck December 30, 2017, 9:58pm 1

```
impl<R: Read, H: Digest> BufRead for PackBufReader<R, H> {
   fn fill_buf(&mut self) -> io::Result<&[u8]> {
       self.inner.fill_buf()
   }

   fn consume(&mut self, amt: usize) {
       {
           //let buf = self.inner.fill_buf().unwrap(); // (1) This line works
           let buf = self.fill_buf().unwrap(); // (2) This does not work

           let len = std::cmp::min(amt, buf.len());
           self.hasher.process(&buf[0..len]);
       }

       self.consumed += amt;
       self.inner.consume(amt)
   }
}
```

Is there any way to annotate `fill_buf()` so that line (2) also works? (I assume line (1) works because rust can see that the fields are disjoint).

Compile log:

```
error[E0499]: cannot borrow `self.hasher` as mutable more than once at a time
  --> src/main.rs:126:13
   |
123 |             let buf = self.fill_buf().unwrap();
   |                       ---- first mutable borrow occurs here
...
126 |             self.hasher.process(&buf[0..len]);
   |             ^^^^^^^^^^^ second mutable borrow occurs here
127 |         }
   |         - first borrow ends here

error: aborting due to previous error
```

1 Like

vitalyd December 30, 2017, 10:25pm 2

Unfortunately there’s no way to annotate or otherwise make it work as written. As you say, the disjointness is “lost” when you call through the function taking `&mut self`. That said, there’s nothing wrong going through the fields to make the compiler see the disjointness. Alternatively, you can pass the fields as references to the helper functions for a similar effect.

2 Likes

Related topics

Topic	Replies	Views	Activity
Cannot borrow mutable self when accessing disjoint fields help	2	571	July 1, 2020
Mutable references help	5	828	September 2, 2020
How to work around multiple mutable borrow of *self? help	3	1169	July 16, 2019
Cannot split mutable borrows of struct fields in poll_flush implementation	5	75	May 31, 2025
Mutability issue help	9	1020	January 12, 2023

- Home