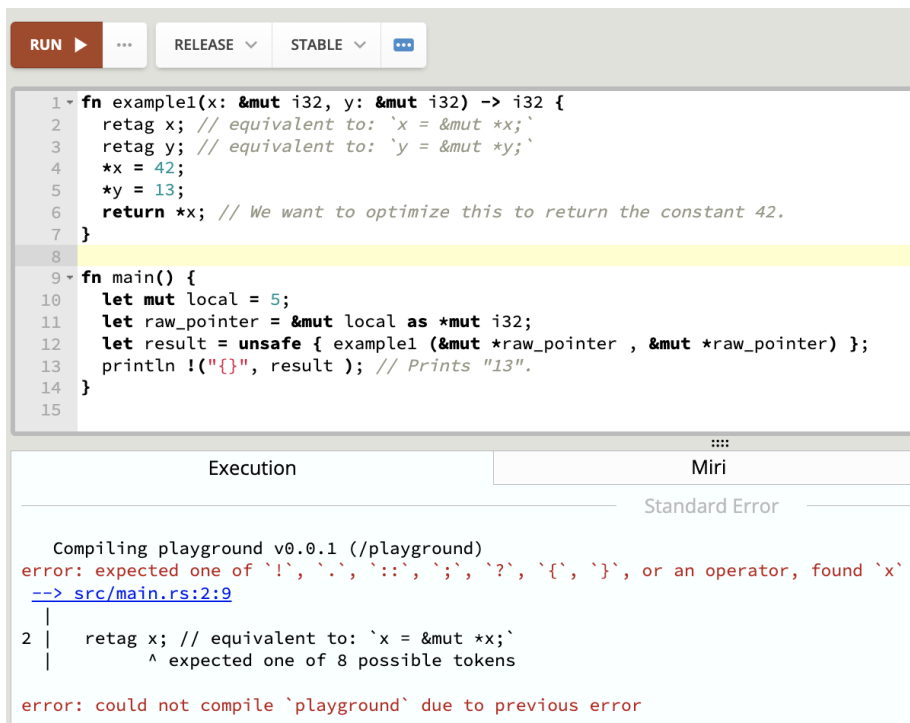


Stacked Borrows: An Aliasing Model for Rust

Here I show the results of running the example codes on gitlab [Stacked Borrows Coq](#):

For example1 in section 3.4, the paper introduces the ‘retag’ identifier. However, when I tried to write ‘retag x’ and ‘retag y’ and run, there would be an error:



```
1 fn example1(x: &mut i32, y: &mut i32) -> i32 {
2   retag x; // equivalent to: `x = &mut *x;`
3   retag y; // equivalent to: `y = &mut *y;`
4   *x = 42;
5   *y = 13;
6   return *x; // We want to optimize this to return the constant 42.
7 }
8
9 fn main() {
10  let mut local = 5;
11  let raw_pointer = &mut local as *mut i32;
12  let result = unsafe { example1 (&mut *raw_pointer , &mut *raw_pointer) };
13  println!("{}", result ); // Prints "13".
14 }
15
```

Execution

Miri

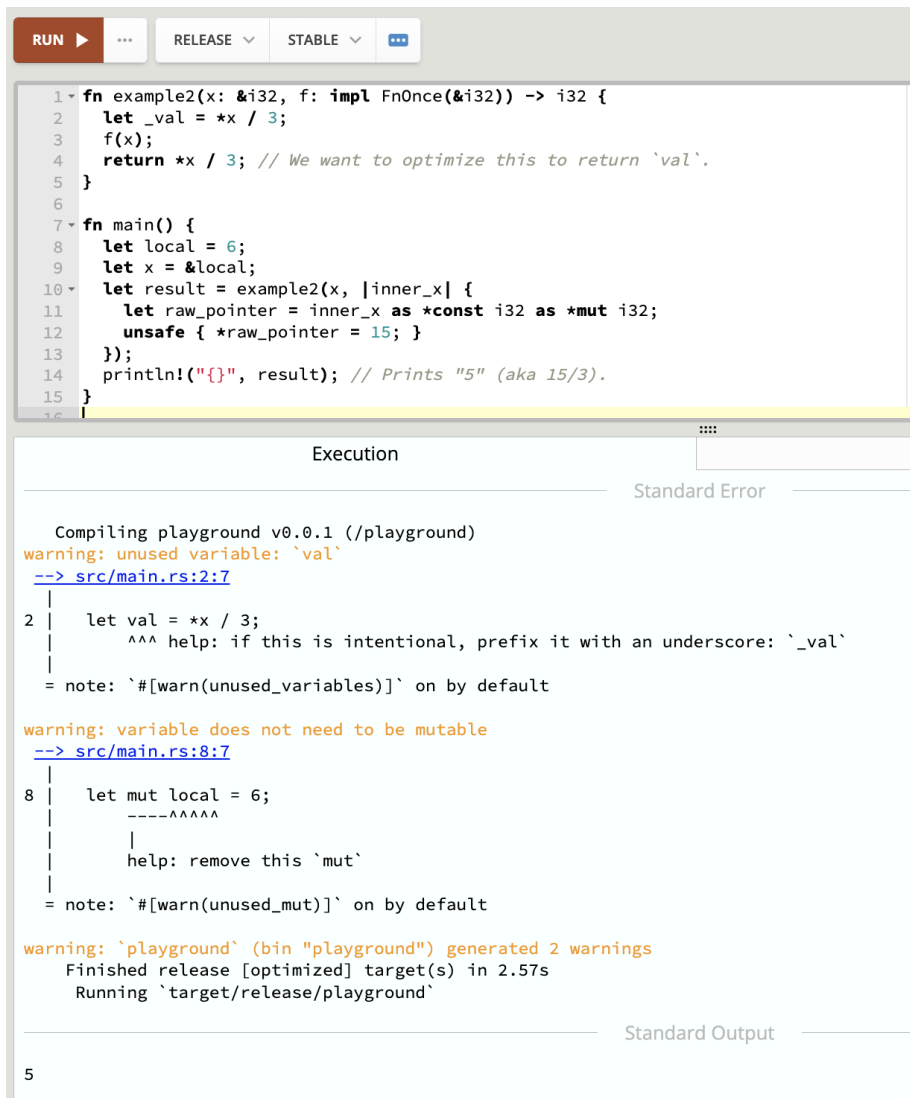
Standard Error

Compiling playground v0.0.1 (/playground)
error: expected one of `!`, `.`, `::`, `;`, `?`, `{`, `}`, or an operator, found `x`
--> [src/main.rs:2:9](#)

```
2 |   retag x; // equivalent to: `x = &mut *x;`
  |           ^ expected one of 8 possible tokens
```

error: could not compile `playground` due to previous error

For example2 in section 3.6, when I tried to run it directly, there would be some errors, although the result '5' showed. After I made the changes as shown in the prompt, the program ran successfully (no error reported).



The screenshot shows a Rust playground interface. At the top, there are buttons for 'RUN', 'RELEASE', and 'STABLE'. Below these is a code editor with the following Rust code:

```
1 fn example2(x: &i32, f: impl FnOnce(&i32)) -> i32 {
2     let _val = *x / 3;
3     f(x);
4     return *x / 3; // We want to optimize this to return `val`.
5 }
6
7 fn main() {
8     let local = 6;
9     let x = &local;
10    let result = example2(x, |inner_x| {
11        let raw_pointer = inner_x as *const i32 as *mut i32;
12        unsafe { *raw_pointer = 15; }
13    });
14    println!("{}", result); // Prints "5" (aka 15/3).
15 }
```

Below the code editor is a section titled 'Execution' with a 'Standard Error' tab. The output shows the following warnings and messages:

```
Compiling playground v0.0.1 (/playground)
warning: unused variable: `_val`
--> src/main.rs:2:7
2 |   let val = *x / 3;
  |     ^^^ help: if this is intentional, prefix it with an underscore: `_val`
= note: `#[warn(unused_variables)]` on by default

warning: variable does not need to be mutable
--> src/main.rs:8:7
8 |   let mut local = 6;
  |     ^^^^^^^^^
  |     help: remove this `mut`
= note: `#[warn(unused_mut)]` on by default

warning: `playground` (bin "playground") generated 2 warnings
Finished release [optimized] target(s) in 2.57s
Running `target/release/playground`
```

At the bottom, there is a 'Standard Output' tab showing the result '5'.