This is a new section to appear at the end of Appendix A, after the “Keywords Reserved for Future Use” section.

Raw Identifiers

Raw identifiers are the syntax that lets you use keywords where they wouldn’t normally be allowed. You use a raw identifier by prefixing a keyword with r#.

For example, match is a keyword. If you try to compile the following function that uses match as its name:

Filename: src/main.rs

fn match(needle: &str, haystack: &str) -> bool {

haystack.contains(needle)

}

you’ll get this error:

error: expected identifier, found keyword `match`

--> src/main.rs:4:4

|

4 | fn match(needle: &str, haystack: &str) -> bool {

| ^^^^^ expected identifier, found keyword

The error shows that you can’t use the keyword match as the function identifier. To use match as a function name, you need to use the raw identifier syntax, like this:

Filename: src/main.rs

fn r#match(needle: &str, haystack: &str) -> bool {

haystack.contains(needle)

}

fn main() {

assert!(r#match("foo", "foobar"));

}

This code will compile without any errors. Note the r# prefix on the function name in its definition as well as where the function is called in main.

Raw identifiers allow you to use any word you choose as an identifier, even if that word happens to be a reserved keyword. In addition, raw identifiers allow you to use libraries written in a different Rust edition than your crate uses. For example, try isn’t a keyword in the 2015 edition but is in the 2018 edition. If you depend on a library that’s written using the 2015 edition and has a try function, you’ll need to use the raw identifier syntax, r#try in this case, to call that function from your 2018 edition code. See Appendix E for more information on editions.