Please insert this text on page 215 between the last two paragraphs before the “Controlling how tests are run” section.

Using Result<T, E> in Tests

So far, we’ve written tests that fail when code within the test panics. We can also write tests that use Result<T, E> and fail when code within the test returns the Err variant! Here’s the test from Listing 11-1, rewritten to use Result<T, E> instead of panicking:

#[cfg(test)]

mod tests {

#[test]

fn it\_works() -> Result<(), String> {

if 2 + 2 == 4 {

Ok(())

} else {

Err(String::from("two plus two does not equal four"))

}

}

}

The it\_works function now has a return type, Result<(), String>. In the body of the function, rather than calling the assert\_eq! macro, we return Ok(()) when the test passes and an Err with a String inside when the test fails. Writing tests that return a Result<T, E> enables you to use the ? operator in the body of tests, which can be a convenient way to write tests that should fail if any operation within them returns an Err variant.

You can’t use the #[should\_panic] annotation on tests that use Result<T, E>. Instead, you should return an Err value directly when the test should fail.